INSA ALBERT EINSTEIN PROFESSORSHIP

PUBLICATIONS

G.N. Ramachandran

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

VOLUME - IX MR 60 - 64

VOLUME - IX   MR 60 - 64

CONTENTS

MR No.	Title	No. of Pages.
60	MATLOG Program in FORTRAN Part I : Summary of the subprograms BVMF and SNSLOG.	41
61	MATLOG Program in FORTRAN Part II : MATLOG Program for Quantifier Logic and their tests.	68
62	MATLOG Program FORTRAN Part III : PROGRAM for GLOGIC and the theory of relations in general.	48
63	Stereochemistry of COLLAGEN. (Invited Review Article)	29
64	The Age of Computerization in Science.	41

MATLOG Program in FORTRAN

Part I : Summary of the subprograms BVMF and SNSLOG

G.N. Ramachandran
INSA Albert Einstein Professor

and

T.A. Thanaraj*
Visiting Fellow

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

*(Permanent Address: Centre for Cellular and Molecular Biology,
Hyderabad 500 007.)

Matphil Reports No. 60, June 1987

(Revised and enlarged draft of ALOG-54, May 1987)

MATLOG Program in FORTRAN

Part I : Summary of the subprograms BVMF and SNSLOG

G.N. Ramachandran
INSA Albert Einstein Professor

and

T.A. Thanaraj*
Visiting Fellow

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

*(Permanent Address: Centre for Cellular and Molecular
Biology, Hyderabad 500 007)

MR-60, June 1987
(Revised and enlarged draft of ALOG-54, May 1987)

Preface

This series of reports, entitled MATLOG Program in FORTRAN, summarizes the formulation of BVMF in a practical manner in the form of FORTRAN statements of our BVMF techniques. They closely follow the theoretical approaches developed in Lecture Series 2 and 3. The first three parts will deal with the subprograms that are required for working out atomic statements in SNS logic, quantifier logic, and general logic dealing with the theory of relations.

Part I contains two subprograms BVMF and SNSLOG dealing respectively with the basic notation and algorithms of BVMF in general, and their application to BA-2 algebra as adopted for SNS logic. Part II will contain the extension of this to BA-3 algebra required for quantifier logic, and Part III to general m x n matrices required for multivalued logic adopted for the general theory of relations.

It is believed that the subroutines included in these reports are fairly exhaustive and are sufficient to work out problems in propositional calculus and in predicate calculus. A number of examples are worked out, using the program. In fact, the conclusions derived from these examples are interesting and the computer-calculated outputs have even led to further refinements of the basic Boolean algebraic theory, via the vector-matrix formalism, of logic.

CONTENTS

Page No.

1. BVMF

(a) BA-1 algebra 1

(b) Vectors 2

(c) Matrices 2

(d) Constants in SNS and QL algebra 3

(e) Logical connectives in SNS and QL 3

2. SNSLOG

(a) Algebraic operators 4

(b) Logical operators 8

3. Checking of the subroutine in practical problems 14

Problem 1A 14

Conclusions made from the output of Problem 1A 15

Problem 1B 17

Comment on the output of Problem 1B 17

Problem 2 19

Solution of Problem 2 20

Problem 3 21

Solution of Problem 3 22

4. Additional problems in SNSLOG 22

Problem 4A 24

Comments on the output of Problem 4A 26

Problem 4B 37

Comments on Problem 4B 38

Problem 4C 38

Comments on Problem 4C 40

Summary of FORTRAN formulae for the program MATLOG

The program will consist initially of four parts, namely BVMF, SNSLOG, QLOGIC and GLOGIC.

1. BVMF

(a) BA-1 algebra

In MATLOG we use only Boolean variables and constants having the two values 1, 0 denoted by the symbols B1, B2. Variables in BA-1 algebra will be denoted by BA, BB, etc., Four basic functions are employed, namely BSUM, BPDT, BEQU, BCMP corresponding to the binary relations $a \oplus b = c$, $a \otimes b = c$, $a \Leftrightarrow b = c$, and the unary relation $a^c = b$ respectively. The essence of the FORTRAN formulae for these is as follows:

Boolean sum

```
          BSUM (BA, BB)  =  BA + BB

          IF  BA.AND.BB  =  1 ,  BC = 1
```

Boolean product

```
          BPDT (BA, BB)  =  BA * BB
```

Boolean equivalence   BEQU(BA,BB) = BC

```
          IF  BA.EQ.BB ,  BC = 1
          ELSE  BC =  0
```

Boolean complement   BCMP(BA) = BB

```
               BB  =  1 - BA
```

(b) Vectors

Vectors are indicated in general by GVA,M , GVB,M , etc where G stands for "general", V stands for "vector" and M stands for the number of components in the vector . Thus, GVA(I) , I = 1, MI are the components of GVA,MI .

For the particular cases of MI = 2 and 3 (namely SNS algebra and quantifier algebra), the vectors are denoted by SVA and QVA for the two cases of MI = 2, 3 and MI is not explicitly mentioned, but is indicated by the letters S and Q at the beginning of the/name of the variable, or function, as the case may be.

(c) Matrices

These are indicated by GMZ,M,N etc., where G denotes a general M x N matrix. the letter M stands for "matrix" and Z is the name of the matrix. The parameters M, N are the dimensions of the matrix so that

GMZ(I,J) , I = 1, M, J = 1, N

represent the components of a general matrix GMZ,M,N

In this case also, if G is replaced by S or Q, then M = N = 2 or 3 as the case may be, and the matrix refers to SNS algebra and quantifier algebra respectively.

(d) Constants in SNS and QL algebra

There are four constants (truth values) in SNS indicated by S1, S2, S3, S4 standing for the vectors (1 0), (0 1), (1 1), (0 0) respectively, for the truth values T, F, D, X respectively.

Similarly, in quantifier algebra, Q1 to Q8 will represent the eight possible quantifier states (constants) in BA-3 algebra. The convention we shall follow is what has been adopted in our BVMF as given below.

Logical symbol	FORTRAN symbol	Name	BA-3 vector	Logical symbol	FORTRAN symbol	Name	BA-3 vector
∀	Q1	'ALL'	(1 0 0)	Φ	Q5	'NEX'	(0 0 1)
Λ	Q2	'NAL'	(0 1 1)	∃	Q6	'EXS'	(1 1 0)
Σ	Q3	'SOM'	(0 1 0)	Δ	Q7	'IND'	(1 1 1)
Θ	Q4	'AON'	(1 0 1)	∅	Q8	'IMP'*	(0 0 0)

*This will be changed to 'XXX' in Part IV .

(e) Logical connectives in SNS and QL

The ten possible logical connectives in SNS are denoted by SMA,SK,SL; SMO,SK,SL; SME,SK,SL; standing respectively for the matrices $\underline{\underline{A}}(k, \ell)$, $\underline{\underline{O}}(k, \ell)$, $\underline{\underline{E}}(k, \ell)$, $k, \ell = 1, 4$. A similar notation is adopted in QLOGIC, namely

QMA,QK,QL; QMO,QK,QL; QME,QK,QL; QM2,QK,QL standing for $\underset{\sim}{Z}(\underset{\sim}{k}, \underset{\sim}{\ell})$,
$\underset{\sim}{k}, \underset{\sim}{\ell} = 1, 8$

The subprograms in SNS for developing these matrices are given in the appropriate sections below. There are four different A's, and four different O's, while only two different E's are present, making in all the ten possible logical connectives.

2. SNSLOG

(a) Algebraic operators

In this section, we shall write the various functions which are of value in SNS logic. They are given names that are suitable for both logical as well as algebraic purposes, and a general notation extendable to QLOGIC and GLOGIC is followed. However, the idea is that for M = N = 2, or 3, the subroutines are written for these values as such without generalizing them to all values of M and N. These simple subroutines define the functions in these two subprograms and they will be generalized later for GLOGIC. Those required in SNSLOG are as follows.

(i) Boolean sum of two vectors $\underline{\underline{a}} \oplus \underline{\underline{b}} = \underline{\underline{c}}$ $(\oplus = \underline{\underline{U}} = \text{union})$

```
SUNION (SVA,SVB) = SVC

 BSUM (SVA(1),SVB(1))= SVC(1)

 BSUM (SVA(2),SVB(2))= SVC(2)
```

(ii) Boolean product of two vectors $\underline{\underline{a}} \otimes \underline{\underline{b}} = \underline{\underline{c}}$ $(\otimes = \underline{\underline{V}} = \text{vidya})$

```
SVIDYA(SVA,SVB) = SVC

BPDT(SVA(1),SVB(1))= SVC(1)

BPDT(SVA(2),SVB(2))= SVC(2)
```

(iii) Boolean complement of SNS vector $\underline{\underline{a}}^c = \underline{\underline{b}}$

```
SCOMP(SVA) = SVB
BCMP(SVA(1)) = SVB(1)
BCMP(SVA(2)) = SVB(2)
```

(iv) Boolean sum of two matrices $\underline{\underline{P}} \oplus \underline{\underline{Q}} = \underline{\underline{R}}$

```
                SMXSUM(SMP,SMQ) = SMR
    BSUM(SMP(I,J), SMQ(I,J)) = SMR(I,J), I, J = 1, 2
```

(v) Boolean product of two matrices $\underline{\underline{P}} \otimes \underline{\underline{Q}} = \underline{\underline{R}}$

Replace SMXSUM in (iv) by SMXPDT, and BSUM by BPDT

(vi) Boolean complement of a matrix operator $\underline{\underline{P}}^c = \underline{\underline{Q}}$

```
SMXCMP(SMP) = SMQ
BCMP(SMP(I)) = SMQ(I), I = 1, 2
```

(vii) Scalar product of two vectors $\langle a | b \rangle = c = a_1 \otimes b_1 \oplus a_2 \otimes$

```
SSCPDT(SVA,SVB) = BC      as above.
BC = BPDT(SVA(1),SVB(1))
BC = BSUM(BC,BPDT(SVA(2),SVB(2))
```

(viii) Unary product of a vector with a matrix

$$\langle a | Z | = \langle b | , \quad \bigoplus_i a_i \otimes Z_{ij} = b_j$$

```
SUNPDT(SVA,SMZ) = SVB
SVB(J) = 0
Do I,J = 1, 2
SVB(J) = BSUM(SVB(J),BPDT(SVA(I),SMZ(I,J)))
```

(This is used for finding the output of a unary relation in SNS. In BVMF, in general, it is the matrix product of a 2-vector with a 2x2 matrix.)

(ix) Binary product of a matrix with two vectors $\langle a|Z|b\rangle = c$

```
SBINPT(SVA,SMZ,SVB) = BC
SVAP = SUNPDT(SVA,SMZ)
BC   = SSCPDT(SVAP,SVB)
```

(x) Matrix product of two matrices : $\bigoplus_{j} P_{ij} \otimes Q_{jk} = R_{ik}$

```
                SMATPT(SMP,SMQ) = SMR
                SMR(I,K) = 0
                DO  I,J,K = 1, 2
        SMR(I,K) = BSUM(SMR(I,K), BPDT(SMP(I,J), SMQ(J,K))
```

(This is used for the successive application of implications in particular and of matrix operations in general).

(xi) Direct product of two vectors: $\underline{\underline{a}} \times \underline{\underline{b}} = \underline{\underline{Z}}$

```
SDIRPT(SVA,SVB) = SMZ
SMZ(I,J) = BPDT(SVA(I),SVB(J)), I,J = 1,2
```

(xii) Direct sum of two vectors: $\underline{\underline{a}} + \underline{\underline{b}} = \underline{\underline{Z}}$

```
SDIRSM(SVA,SVB) = SMZ
SMZ(I,J) = BSUM(SVA(I),SVB(J)), I,J = 1,2
```

(xiii) Direct equivalence of two vectors: $(\underline{\underline{a}} \equiv \underline{\underline{b}}) = \underline{\underline{Z}}$

```
SDIREQ(SVA,SVB) = SMZ
SMZ(I,J) = BEQU(SVA(I),SVB(J)), I,J = 1,2
```

(xiv) Scalar vector direct product $a \times \underline{\underline{b}} = \underline{\underline{c}}$

```
SSVDPT(BA,SVB)  =  SVC
SVC(I) = BPDT(BA,SVB(I)), I = 1, 2
```

(d) Constants in SNS and QL algebra

There are four constants (truth values) in SNS indicated by S1, S2, S3, S4 standing for the vectors (1 0), (0 1), (1 1), (0 0) respectively, for the truth values T, F, D, X respectively.

Similarly, in quantifier algebra, Q1 to Q8 will represent the eight possible quantifier states (constants) in BA-3 algebra. The convention we shall follow is what has been adopted in our BVMF as given below.

Logical symbol	FORTRAN symbol	Name	BA-3 vector	Logical symbol	FORTRAN symbol	Name	BA-3 vector
$\forall$	Q1	'ALL'	(1 0 0)	Φ	Q5	'NEX'	(0 0 1)
Λ	Q2	'NAL'	(0 1 1)	$\exists$	Q6	'EXS'	(1 1 0)
Σ	Q3	'SOM'	(0 1 0)	Δ	Q7	'IND'	(1 1 1)
Θ	Q4	'AON'	(1 0 1)	ϕ	Q8	'IMP'*	(0 0 0)

*This will be changed to 'XXX' in Part IV .

(e) Logical connectives in SNS and QL

The ten possible logical connectives in SNS are denoted by SMA,SK,SL; SMO,SK,SL; SME,SK,SL; standing respectively for the matrices $\underline{\underline{A}}(k, \ell)$, $\underline{\underline{O}}(k, \ell)$, $\underline{\underline{E}}(k, \ell)$, $k, \ell = 1, 4$. A similar notation is adopted in QLOGIC, namely

QMA,QK,QL; QMO,QK,QL; QME,QK,QL; QM2,QK,QL standing for $\underset{\sim}{Z}(\underset{\sim}{k}, \underset{\sim}{\ell})$,
$\underset{\sim}{k}, \underset{\sim}{\ell} = 1, 8$

The subprograms in SNS for developing these matrices are given in the appropriate sections below. There are four different A's, and four different O's, while only two different E's are present, making in all the ten possible logical connectives.

"general" logic. However, in GLOGIC, we have to specify also the values of M and N for matrices and vectors. The list of these *Boolean algebraic* formulae will be given in Parts 2 and 3, and they follow very similar patterns in QLOGIC and GLOGIC as those stated above for SNSLOG. However, in the application of these formulae to obtain various results of significance in logic *per se*, as distinct from algebra in BVMF, the formulae are slightly different for the three cases of SNS (BA-2), QL(BA-3) and general logic (BA-n). This is particularly so for the functions and definitions given below in subsection (b).

(b) Logical operators

(xvii) Logical connectives $\underline{\underline{A}}(k, \ell)$, $\underline{\underline{O}}(k, \ell)$, $\underline{\underline{E}}(k, \ell)$, $\underline{\underline{I}}(k, \ell)$

In an obvious notation these are representable in FORTRAN as follows.

```
SMA,SK,SL  =  SDIRPT(SK, SL)
SMO,SK,SL  =  SDIRSM(SK, SL)
SME,SK,SL  =  SDIREQ(SK, SL)
SMI,SK,SL  =  SDIRSM(SCOMP(SK), SL)
```

These formulae for the standard logical connectives for conjunction, disjunction, equivalence or negation, and implication, require only K and L equal to 1 or 2. However, they can be analytically continued, for K,L = 3,4 also, for $\underline{\underline{A}}(k, \ell)$ and $\underline{\underline{O}}(k, \ell)$, and hence to $\underline{\underline{E}}(k, \ell)$ and $\underline{\underline{I}}(k, \ell)$, which covers all sixteen possible 2x2 Boolean matrices, as discussed in Lecture-2 Series-3, (see Tables 2 and 3.)

(xviii) Relative truth value of one term for another :

$$\underline{\underline{t}}(\underline{\underline{a}} \mid \underline{\underline{b}}) = (\langle a \mid b \rangle \ , \langle a \mid b^c \rangle) = (c_1, c_2)$$

```
SRELTV(SVA, SVB)  =  SVC
SVC(1) = SSCPDT(SVA, SVB)
SVC(2) = SSCPDT(SVA, SCOMP(SVB))
```

The four truth values for $\underline{\underline{c}}$, namely T, F, D, X, correspond to the following inclusion properties of the vectors $\underline{\underline{a}}$ and $\underline{\underline{b}}$.

T ⟷ $\underline{\underline{a}}$ is fully included in $\underline{\underline{b}}$, and has no intersection with $\underline{\underline{b}}^c$.

F ⟷ $\underline{\underline{a}}$ is fully included in $\underline{\underline{b}}^c$ and is not included in $\underline{\underline{b}}$.

D ⟷ $\underline{\underline{a}}$ is partially included in $\underline{\underline{b}}$ and is also partially included in $\underline{\underline{b}}^c$.

X ⟷ $\underline{\underline{a}}$ is the null vector X = (0 0), which is not included in either $\underline{\underline{b}}$ or $\underline{\underline{b}}^c$

(xix) SNS truth value of the binary relation $\underline{\underline{Z}}$ for inputs $\underline{\underline{a}}$, $\underline{\underline{b}}$ (formula via contracted product) for a general 2x2 matrix $\underline{\underline{Z}}$

$$\underline{\underline{t}}(\underline{\underline{a}}\ \underline{\underline{Z}}\ \underline{\underline{b}}) = (\underline{\underline{a}}|\underline{\underline{Z}}|\underline{\underline{b}}) = \underline{\underline{c}} = (c_1, c_2) ,$$

$$\text{where}\quad c_1 = \langle \underline{\underline{a}}|\underline{\underline{Z}}|\underline{\underline{b}}\rangle, \quad c_2 = \langle \underline{\underline{a}}|\underline{\underline{Z}}^c|\underline{\underline{b}}\rangle$$

```
SSTV(SVA, SMZ, SVB) = SVC

SVC(1) = SBINPT(SVA, SMZ, SVB)

SVC(2) = SBINPT(SVA, SCOMP(SMZ), SVB)
```

This function SSTV is the general formula for the truth value of a SNS logical relation when we are given the four components of the 2x2 matrix $\underline{\underline{Z}}$ corresponding to the connective, without specifying its logical nature as being 'and', 'or', 'equ', 'imp' etc. This has been used in the previous programs, and is generally valid also for QLOGIC and GLOGIC for all values of M, N.

However, a variation of this function is given below as SSTV2, which can be used when the logical nature of the connective is specified as $\underline{\underline{Z}}(k, \ell)$, with $\underline{\underline{Z}} = \underline{\underline{A}}, \underline{\underline{O}}, \underline{\underline{E}}$, or $\underline{\underline{I}}$, as the case may be. The corresponding 2x2 matrices are all direct sums, direct products, or direct equivalences, of two 2-vectors, and therefore the formulae are simpler to manipulate, **particularly in GLOGIC (see later).**

(xx) SNS truth value of a relation involving the connective $\underline{\underline{Z}}(k, \ell)$ for inputs $\underline{\underline{a}}$, $\underline{\underline{b}}$ (See Lecture 4,Series 2, p.1 of MR-

```
      SSTV2(SVA, SMZ, SVB) = SVC

IF  SMZ = SMA,SK,SL
      SVC(1) = BPDT(SSCPDT(SVA, SK),SSCPDT(SVB, SL))
      SVC(2) = BSUM(SSCPDT(SVA, SCOMP(SK)),SSCPDT(SVB, SCOMP(SL))

IF  SMZ = SMO,SK,SL
      SVC(1) = BSUM(SSCPDT(SVA, SK),SSCPDT(SVB, SL))
      SVC(2) = BPDT(SSCPDT(SVA, SCOMP(SK)),SSCPDT(SVB, SCOMP(SL))

IF  SMZ = SME,SK,SL
      SKP = SCOMP(SK), SLP = SCOMP(SL)
      SVCA = SSTV2(SVA, SMA,SK,SL, SVB)
      SVCB = SSTV2(SVA, SMA,SKP,SLP, SVB)
      SVC(1) = BSUM(SVCA(1),SVCB(1))
      SVC(2) = BPDT(SVCA(2),SVCB(2))

IF  SMZ = SMI,SK,SL
      SKP = SCOMP(SK)
      SMZ = SMO,SKP,SL
```

This subroutine is very much more in the spirit of orthodox logic, and should preferably be used for straightforward logical connectives of the type mentioned in the subroutine. However, both in SNS as well as in QL, general 2x2 matrices and 3x3 matrices can occur for relations and the previous subrout SSTV in (xix) will have to be invoked in such cases.

(xxi) Output of unary relation for $\underline{\underline{Z}}(k, \ell)$ via relative truth value: $\langle \underline{\underline{a}} | s(k) \rangle \; Z \; \langle s(\ell) | = \underline{\underline{b}}$, $Z = \otimes$ or $\oplus$

```
            SUNPT2(SVA,SMZ)  =  SVB

IF   SMZ  =  SMA,SK,SL
            BAP  =  SSCPDT(SVA,SK)
            SVB  =  SSVDPT(BAP,SL)

IF   SMZ  =  SMO,SK,SL
            BAP  =  SSCPDT(SVA,SK)
            SVB  =  SSVDSM(BAP,SL)

IF   SMZ  =  SMI,SK,SL
            BAP  =  SSCPDT(SVA,SCOMP(SK))
            SVB  =  SSVDSM(BAP,SL)

IF   SMZ  =  SME,SK,SL
            SKP  =  SCOMP(SK),   SLP  =  SCOMP(SL)
           SVBA   =  SUNPT2(SVA,  SMA,SK,SL)
           SVBB   =  SUNPT2(SVA,  SMA,SKP,SLP)
            SVB  =  SUNION(SVBA,SVBB)
```

(xxii) Matrix-Boolean binary operator "agree" $\underline{\underline{G}}$, for checking agreement of two SNS terms : $(\underline{\underline{a}}|\underline{\underline{G}}|\underline{\underline{b}}) = \underline{\underline{c}}$

```
            SMBG(SVA,SVB)  =  SVC

            BCA  =  BEQU(SVA(1),SVB(1))
            BCB  =  BEQU(SVA(2),SVB(2))

           SVG(1)  =  BPDT(BCA,BCB)
           SVG(2)  =  BCMP(SVG(1))
```

(xxiii) Matrix-Boolean unary operators $\underline{\underline{E}}$, $\underline{\underline{N}}$, $\underline{\underline{M}}$, $\underline{\underline{L}}$

These are simple operators, defined analogously to quantifier logic, which have been found to be useful in SNS algebra also. In an obvious notation these are

```
SEQU(SVA) = SVB
SVB(1) = SVA(1), SVB(2) = SVA(2)

SNOT(SVA) = SVB
SVB(1) = SVA(2), SVB(2) = SVB(1)

SCOMP(SVA) = SVB
SVB(1) = BCMP(SVA(1)), SVB(2) = BCMP(SVA(2))

SELL(SVA) = SVB
SVB(1) = BCMP(SVA(2)), SVB(2) = BCMP(SVA(1))
```

Three of these operators are equivalent to those already defined earlier, namely

SEQU ⟺ SME,S1,S1, SNOT ⟺ SME,S1,S2, SCOMP ⟺ (ii)

However, the adoption of the names given here, employing simpler definitions given above, makes the implementation of many problems simpler. In fact, even the analogous functions QEQU and QNOT are not definable in terms of QMZ,QK,QL in quantifier logic.

3. Checking of the subroutine in practical problems

The above list of subroutines in SNSLOGIC appear to be sufficient for all applications. A few problems are given below, including routine testing of subroutines, which will both check these, as well as illustrate their applications. In particular, the subroutines (i) to (xvi) for BVMF algebra are not checked as such, but their application in subroutines (xvii) to (xxiii) for SNSLOGIC is tested by working out truth tables and so on for logical derivations.

Problem 1A

Work out the 4x4 truth tables for the SNS truth values T, F, D, X for $\underline{a}$ and $\underline{b}$, for the 16 possible matrix connectives each, of the types $\underline{\underline{A}}(k, \ell)$, $\underline{\underline{O}}(k, \ell)$, $\underline{\underline{E}}(k, \ell)$ and $\underline{\underline{I}}(k, \ell)$. Do this by using

(a) subroutine (xix) — SSTV
(b) subroutine (xx) — SSTV2

and check the equivalence of each table obtained by employing the two functions SSTV and SSTV2.

Print out the 4x4 truth values for the latter (SSTV2) for all of them and note the differences if any between SSTV and SSTV2.

Conclusions made from the output of Problem 1A

On checking for discrepancies between the 4x4 truth tables, it was found that the 3x3 tables for the states T, F and D were completely in agreement, while there could be disagreement if either of the inputs s(a) or s(b) is X. (See Table 1 in the next page 16 for typical examples.) It will be seen from the table that, when either a or b is X , the output truth value is also X, uniformly for all connectives, with SSTV. On the other hand, it could also be either T or F with SSTV2. These two correspond to two different interpretations of the input state X = (0 0), namely, as corresponding to "absent information" for SSTV2, and "contradictory information" for SSTV. (This will become still clearer when we discuss QLOGIC, where the differences between BA-3 states being used for describing the eight quantifier truth values , and for describing multi-component sets of upto three terms by 3-vectors $(a_1\ a_2\ a_3)$, becomes prominent. In the former case, ϕ indicates an impossible truth value, while in the latter case, it only indicates the non-presence of all the three terms in the set under consideration

Table 1A Differences between SSTV and SSTV2

Only the tables for $\underline{\underline{A}}(1, 1)$, $\underline{\underline{O}}(1, 1)$, $\underline{\underline{E}}(1, 1)$ and $\underline{\underline{I}}(1, 1)$ are given below, but the trend for all $\underline{\underline{Z}}(k, \ell)$ is reasonably clear. (See Part 2)

	SSTV				SSTV2			
$\underline{\underline{a}}$ \ $\underline{\underline{b}}$	T	F	D	X	T	F	D	X
T	T	F	D	X	T	F	D	X
F	F	F	F	X	F	F	F	Ⓕ
D	D	F	D	X	D	F	D	Ⓕ
X	X	X	X	X	X	Ⓕ	Ⓕ	X

$\underline{\underline{A}}(1, 1) = \underline{\underline{a}} \wedge \underline{\underline{b}}$

Ⓕ if $\underline{\underline{a}} \supseteq s(2), \underline{\underline{b}}=X$
or $\underline{\underline{b}} \supseteq s(2), \underline{\underline{a}}=X$

$\underline{\underline{a}}$								
T	T	T	T	X	T	T	T	Ⓣ
F	T	F	D	X	T	F	D	X
D	T	D	D	X	T	D	D	Ⓣ
X	X	X	X	X	Ⓣ	X	Ⓣ	X

$\underline{\underline{O}}(1, 1) = \underline{\underline{a}} \vee \underline{\underline{b}}$

Ⓣ if $\underline{\underline{a}} \supseteq s(1), \underline{\underline{b}}=X$
or $\underline{\underline{b}} \supseteq s(1), \underline{\underline{a}}=X$

$\underline{\underline{a}}$								
T	T	F	D	X	T	F	D	X
F	F	T	D	X	F	T	D	X
D	D	D	D	X	D	D	D	Ⓕ
X	X	X	X	X	X	X	Ⓕ	X

$\underline{\underline{E}}(1, 1) = \underline{\underline{a}} \Longleftrightarrow \underline{\underline{b}}$

This has to be checked further after the two ways of defining $\underline{\underline{E}}(k,\ell$ in Problem 4B,C are worked out.

$\underline{\underline{a}}$								
T	T	F	D	X	T	F	D	X
F	T	T	T	X	T	T	T	Ⓣ
D	T	D	D	X	T	D	D	Ⓣ
X	X	X	X	X	Ⓣ	X	Ⓣ	X

$\underline{\underline{I}}(1, 1) = \underline{\underline{a}} \Longrightarrow \underline{\underline{b}}$
$= \neg\underline{\underline{a}} \vee \underline{\underline{b}}$

Ⓣ if $\underline{\underline{a}} \supseteq s(2), \underline{\underline{b}}=X$
or $\underline{\underline{b}} \supseteq s(1), \underline{\underline{a}}=X$

Problem 1B

Similarly, compare the outputs of SUNPDT and SUNPT2 for the unary relation $\underline{a}\ \underline{Z} = \underline{b}$ for all the connectives $\underline{Z}(k, \ell)$ $k, \ell = 1$ to 4, for $\underline{Z} = \underline{A}, \underline{O}, \underline{I}, \underline{E}$.

Comment on the output of Problem 1B

Just as for the comparison between SSTV and SSTV2, in this case also, the output $\underline{b}$ of the two subroutines SUNPDT and SUNPT2 agree, for the three inputs $\underline{a} =$ T, F, D, for all connectives $\underline{Z}(k, \ell)$, but for X as input, they do not agree sometimes. Some typical examples are given in Table 1B.

It will be seen that the two agree for $\underline{A}(k, \ell)$ and $\underline{E}(k, \ell)$ and this is true quite generally. However, for $\underline{O}(k, \ell)$ and $\underline{I}(k, \ell)$, for X as input, SUNPDT gives X as output, while SUNPT2 always gives $s(\ell)$ as output. This is also explicable by the same considerations as mentioned above for Problem 1A.

We shall give in problems 2 and 3 the way in which a set of sequentially implementable BVMF equations can be worked out in FORTRAN for MATLOG. The examples are taken from previous lectures.

Table 1B: Differences between SUNPDT and SUNPT2

Chosen examples are given below and a summary of the observations is given in the text. The first columns correspond to SUNPDT and the second to SUNPT2

	A(1, 1)		A(1, 2)		A(1, 3)		A(1, 4)	
T	T	T	F	F	D	D	X	X
F	X	X	X	X	X	X	X	X
D	T	T	F	F	D	D	X	X
X	X	X	X	X	X	X	X	X

	O(1, 1)		O(1, 2)		O(1, 3)		O(1, 4)	
T	D	D	D	D	D	D	D	D
F	T	T	F	F	D	D	X	X
D	D	D	D	D	D	D	D	D
X	X	Ⓣ	X	Ⓕ	X	Ⓓ	X	X

	O(2, 1)		O(2, 2)		O(2, 3)		O(2, 4)	
T	T	T	F	F	D	D	X	X
F	D	D	D	D	D	D	D	D
D	D	D	D	D	D	D	D	D
X	X	Ⓣ	X	Ⓕ	X	Ⓓ	X	X

	I(1, 1)		I(1, 2)		I(1, 3)		I(1, 4)	
T	T	T	F	F	D	D	X	X
F	D	D	D	D	D	D	D	D
D	D	D	D	D	D	D	D	D
X	X	Ⓣ	X	Ⓕ	X	Ⓓ	X	X

	E(1, 1)		E(1, 2)		E(1, 3)		E(1, 4)	
T	T	T	F	F	D	D	X	X
F	F	F	T	T	X	X	D	D
D	D	D	D	D	D	D	D	D
X	X	X	X	X	X	X	X	X

Problem 2

Take the argument given in Lecture-3, Series-3, page 20.

The formulae in BVMF and in FORTRAN for MATLOG are given below.

The output data are to be printed out in the format given.

	T, F for INPUT/SVA, SVB, SVC
1. $\underline{\underline{a}}\ \underline{\underline{A}}\ \underline{\underline{b}} = \underline{\underline{g}}$	SVG = SSTV2(SVA, SMA,S1,S1, SVB)
2. $\underline{\underline{c}}\ \underline{\underline{O}}\ \underline{\underline{b}} = \underline{\underline{h}}$	SVH = SSTV2(SVC, SMO,S1,S1, SVB)
3. $T = \underline{\underline{x}}''$	SVXPP = S1
4. $\underline{\underline{g}}\ \underline{\underline{I}} = \underline{\underline{k}}$	SVK = SUNPDT(SVG, SMI,S1,S1)
5. $\underline{\underline{h}}\ \underline{\underline{I}}(1,2) = \underline{\underline{y}}$	SVY = SUNPDT(SVH, SMI,S1,S2)
6. $\underline{\underline{h}}\ \underline{\underline{A}}\ \underline{\underline{k}} = \underline{\underline{x}}'$	SVXP = SSTV2(SVH, SMA,S1,S1, SVK)
7. $\underline{\underline{x}}'\ \underline{\underline{V}}\ \underline{\underline{x}}'' = \underline{\underline{x}}$	SVX = SVIDYA(SVXP,SVXPP)
	PRINT SVG, SVH, SVXPP, SVK, SVY, SVXP, SVX

Sl. No.	A	B	C	G	H	XPP	K	Y	XP	X
1.	T	T	T							
		—								
		—								
8.	F	F	F							

Solution of Problem 2

The output as given bv the program for this problem is given below. It can be verified that it agrees completely with the table given in Lecture-3, Series-3 for this problem.

```
A   B   C     G   H    XPP   K   Y   XP     X

T   T   T     T   T     T    T   F   T      T
T   T   F     T   T     T    T   F   T      T
T   F   T     F   T     T    D   F   D      T
T   F   F     F   F     T    D   D   F      X
F   T   T     F   T     T    D   F   D      T
F   T   F     F   T     T    D   F   D      T
F   F   T     F   T     T    D   F   D      T
F   F   F     F   F     T    D   D   F      X
```

It so happens that the only output, SVX, is either T,or X, but this is not generally true. Also, note that the state D occurs as an intermediate output, e.g. for SVK,SVY and SVXP, and F occurs for all other intermediate outputs (SVXPP is an input). See the text of Lecture 3, Series 3 for further discussion.

Problem 3

Consider the problem given in pages 34, 36 of Lecture-3,Seri

The FORTRAN formulae are given below and they have already been arranged for sequential implementation,with the six stages as marked.

```
INPUTS   SVA1, SVA2, SVA3, SVA4

SVB1   =  SUNPDT(SVA1, SMI,S1,S1)                    (1.1)
SVC1   =  SSTV(SVA1, SMO,S1,S1, SVA2)                (1.2)
SVB2   =  SUNPDT(SVA3, SMI,S1,S2)                    (1.3)
SVD2P  =  SSTV(SVA3, SMO,S1,S1, SVA4)                (1.4)

SVC2   =  SSTV(SVA2, SMA,S1,S1, SVB1)                (2.1)
SVD1   =  SSTV(SVA3, SMO,S2,S2, SVC1)                (2.2)

SVD2PP =  SSTV(SVD1, SMA,S1,S1, SVB2)                (3.1)
SVE1P  =  SUNPDT(SVC2, SMI,S1,S1)                    (3.2)

SVD2   =  SVIDYA(SVD2P, SVD2PP)                      (4.1)
PRINT  * * * IF SVIDYA = X,   STOP
SVE1PP =  SUNPDT(SVD2, SMI,S1,S1)                    (4.2)

SVE2   =  SSTV(SVA4, SMO,S1,S1, SVD2)                (5.1)
SVE1   =  SVIDYA(SVE1P, SVE1PP)                      (5.2)
PRINT  * * * IF SVIDYA = X,   STOP

SVE    =  SVIDYA(SVE1, SVE2)                         (6.1)
PRINT  * * * IF SVIDYA = X,   STOP
```

The output is to be printed out in the same manner as Problem 2 with spaces between the different stages.

For ready reference, the classical logic equivalents of the MATLOG equations (1.1) to (6.1) are given in CL-1 to CL-11 below.

$$a1 \Rightarrow b1 \tag{CL-1}$$

$$a1 \vee a2 \Leftrightarrow c1 \tag{CL-2}$$

$$a3 \Rightarrow \neg g1 \tag{CL-3}$$

$$a2 \wedge b1 \Leftrightarrow c2 \tag{CL-4}$$

$$\neg a3 \vee \neg c1 \Leftrightarrow d1 \tag{CL-5}$$

$$\neg g1 \Rightarrow \neg b2 \tag{CL-6}$$

$$d1 \wedge b2 \Leftrightarrow d2 \tag{CL-7}$$

$$a3 \vee a4 \Leftrightarrow d2 \tag{CL-8}$$

$$c2 \Rightarrow x \tag{CL-9}$$

$$d2 \Rightarrow x \tag{CL-10}$$

$$a4 \vee d2 \Leftrightarrow x \tag{CL-11}$$

It should be noticed that CL-7 and CL-8 both have the same output $\underline{d2}$, and should therefore be checked for consistency, and similarly, CL-9, 10, 11, all of which have the same output X, should also be tested for consistency.

The logical graph of the CL statements, and their modification in BVMF by including vidya checks, is contained in Fig.1 on page 21b. For further details Lecture-3, Series-3 may be referred to.

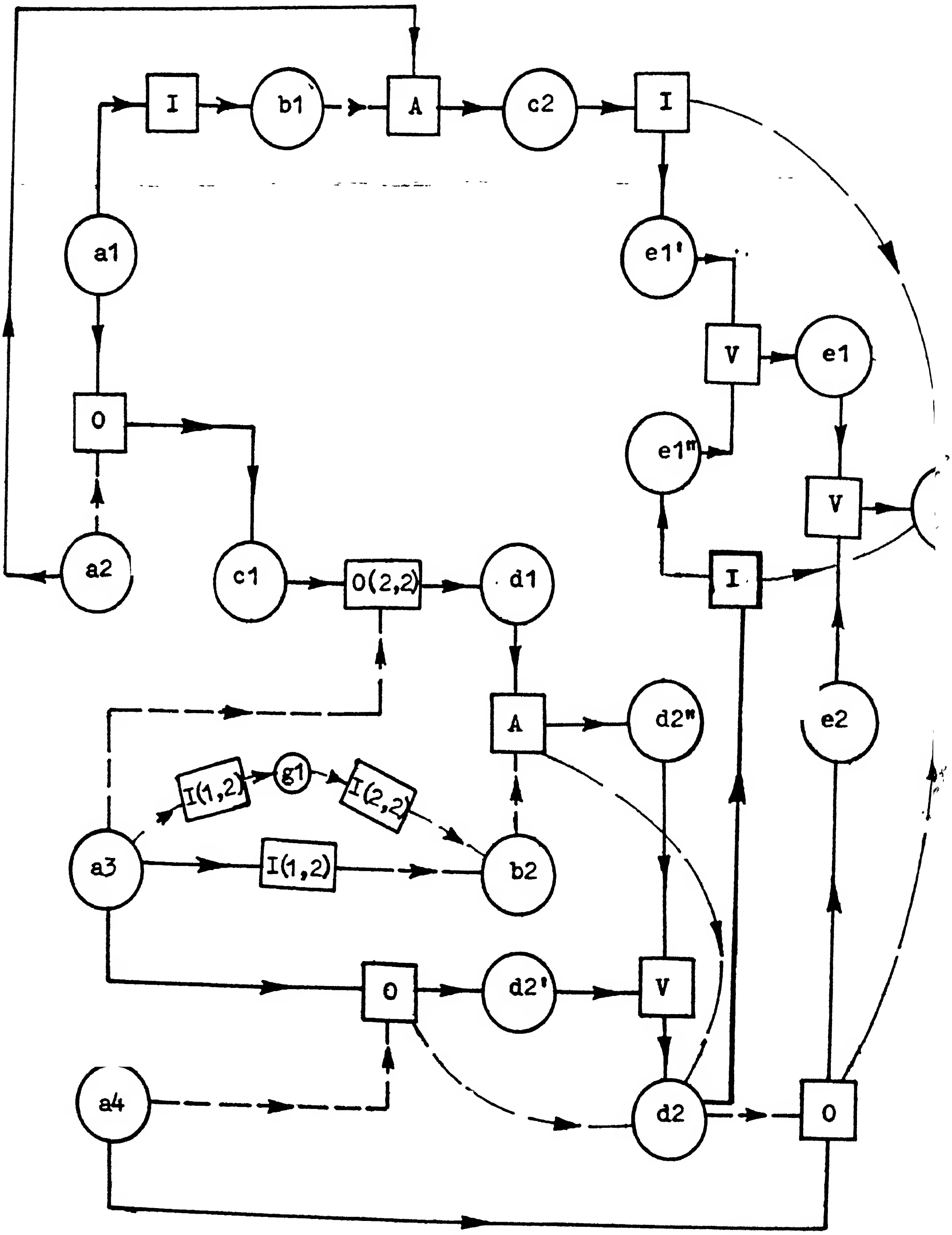


Fig.1. Logical graph corresponding to the set of statements
listed above.

Solution of Problem 3

Just as in the case of Problem 2, for this problem also, the computer output fully substantiates the results worked out manually in Lecture-3, Series-3. The output is given in the next page. There is a small difference in the logical graph in that $\underline{\underline{d2}}$ is the input in (4.2), instead of $\underline{\underline{d2''}}$, and there are minor differences as a consequence, but no changes in the occurrence of contradictions. The predictions made in Lecture-3, as to which of the 16 possible inputs will lead to contradictions, and which will not, are fully verified.

In view of the discussion given above for Problem 1, some additional problems in SNSLOG were worked out and are described below.

4. Additional problems in SNSLOG

A fuller analysis of the differences between SSTV and SSTV2, and between SUNPDT and SUNPT2, will be given in Part-3 while dealing with GLOGIC, after considering also similar differences found in QLOGIC and GLOGIC. In essence, the two

A1	A2	A3	A4	B1	C1	B2	D2P	C2	C1	D2PP	E1P	D2	E1PP	E2	E1	E
T	T	T	T	T	T	F	T	T	F	F	T	***	***	***	***	***
T	T	T	F	T	T	F	T	T	F	F	T	***	***	***	***	***
T	T	F	T	T	T	D	T	T	T	D	T	T	T	T	T	T
T	T	F	F	T	T	D	F	T	T	D	T	F	D	F	T	***
T	F	T	T	T	T	F	T	F	F	F	D	***	***	***	***	***
T	F	T	F	T	T	F	T	F	F	F	D	***	***	***	***	***
T	F	F	T	T	T	D	T	F	T	D	D	T	T	T	T	T
T	F	F	F	T	T	D	F	F	T	D	D	F	D	F	D	F
F	T	T	T	D	T	F	T	C	F	F	D	***	***	***	***	***
F	T	T	F	D	T	F	T	C	F	F	D	***	***	***	***	***
F	T	F	T	D	T	D	T	C	T	D	D	T	T	T	T	T
F	T	F	F	D	T	D	F	C	T	D	D	F	D	F	D	F
F	F	T	T	D	F	F	T	F	T	F	D	***	***	***	***	***
F	F	T	F	D	F	F	T	F	T	F	D	***	***	***	***	***
F	F	F	T	D	F	D	T	F	T	D	D	T	T	T	T	T
F	F	F	F	D	F	D	F	F	T	D	D	F	D	F	D	F

functions are not logically equivalent and therefore, not identical, but they do not lead to any differences in the predictions for classical logic using BA-1 truth values. However, their difference becomes manifest in SNS and QLOGIC employing BA-2 and BA-3 algebra, and particularly so with m x n matrices in GLOGIC. A general discussion is given in Part 3, Section 5 , but we shall consider below two or three examples in the form of problems to illustrate the merits, and consistency, of the matrix formulation in SNS logic. These will be followed up for QLOGIC and later for GLOGIC.

Problem 4A

The following two equations (1) and (2) have been implicitly used in our studies with BVMF so far. A discussion of these will be given in Part-3, which will also indicate their limitations, as well as the conditions under which they are valid.

$$(\underline{\underline{a}}\ \underline{\underline{P}}\ \underline{\underline{b}}) \vee (\underline{\underline{a}}\ \underline{\underline{Q}}\ \underline{\underline{b}}) = \underline{\underline{a}}\ (\underline{\underline{P}} \oplus \underline{\underline{Q}})\ \underline{\underline{b}} \qquad (1)$$

$$(\underline{\underline{a}}\ \underline{\underline{P}}\ \underline{\underline{b}}) \wedge (\underline{\underline{a}}\ \underline{\underline{Q}}\ \underline{\underline{b}}) = \underline{\underline{a}}\ (\underline{\underline{P}} \otimes \underline{\underline{Q}})\ \underline{\underline{b}} \qquad (2)$$

The l.h.s of Eqs (1) and (2) are the disjunction and conjunction respectively of the truth values of the relations $\underline{\underline{a}}\ \underline{\underline{P}}\ \underline{\underline{b}}$ and $\underline{\underline{a}}\ \underline{\underline{Q}}\ \underline{\underline{b}}$, connecting two terms $\underline{\underline{a}}$ and $\underline{\underline{b}}$ via two different relations $\underline{\underline{P}}$ and $\underline{\underline{Q}}$. However, in BVMF, these can be converted into single relations $\underline{\underline{a}}\ \underline{\underline{R}}\ \underline{\underline{b}}$, and $\underline{\underline{a}}\ \underline{\underline{R}}'\ \underline{\underline{b}}$, respectively, with $\underline{\underline{R}} = \underline{\underline{P}} \oplus \underline{\underline{Q}}$ and $\underline{\underline{R}}' = \underline{\underline{P}} \otimes \underline{\underline{Q}}$, and this can be verified, for the inputs T and F for $\underline{\underline{a}}$ and $\underline{\underline{b}}$, in all cases. Obviously, these are the only states that are valid in classical logic, and it would appear as if the extension to SNSLOGIC in Eqs.(1) and (2) would be valid generally. However, a complete test of this, made by employing the subroutines in MATLOG, indicate that this is not always true. This was done by using the following MATLOG equations for checking the validity of Eqs.(1) and (2).

```
SSTV(SVA,SMP,SMB) = SVVP
SSTV(SVA,SMQ,SMB) = SVVQ

SSTV(SVVP,SMO,S1,S1,SVVQ) = SVR1
SMXSUM(SMP,SMQ) = SMR
SSTV(SVA,SMR,SVB) = SVR2
SMBG(SVR1,SVR2) = SVG1

SSTV(SVVP,SMA,S1,S1,SVVQ) = SVR3
SMXPDT(SMP,SMQ) = SMRP
SSTV(SVA,SMRP,SVB) = SVR4
SMBG(SVR3,SVR4) = SVG2
```

The cheoks to be made can be given as follows.

Compare the 4x4 truth tables of SVR1 and SVR2, and of SVR3 and SVR4, for T, F, D, X as inputs for the truth values for SVA and SVB, and print these, as well as the truth values SVG1 and SVG2 of the agreement between these, for the following pairs $\underline{\underline{P}},\underline{\underline{Q}}$:

$\underline{\underline{A}}(1,1),\underline{\underline{A}}(1,2)$; $\underline{\underline{A}}(1,1),\underline{\underline{A}}(2,1)$; $\underline{\underline{A}}(1,1),\underline{\underline{A}}(2,2)$; $\underline{\underline{A}}(1,1),\underline{\underline{A}}(1,3)$;
$\underline{\underline{A}}(3,3),\underline{\underline{A}}(4,4)$;

$\underline{\underline{O}}(2,1),\underline{\underline{O}}(1,2)$; $\underline{\underline{O}}(1,1),\underline{\underline{O}}(2,2)$; $\underline{\underline{O}}(1,1),\underline{\underline{O}}(1,2)$; $\underline{\underline{O}}(3,1),\underline{\underline{O}}(3,2)$;
$\underline{\underline{O}}(1,3),\underline{\underline{O}}(1,4)$;

$\underline{\underline{E}}(1,1),\underline{\underline{E}}(1,2)$; $\underline{\underline{E}}(1,3),\underline{\underline{E}}(3,2)$; $\underline{\underline{E}}(1,4),\underline{\underline{E}}(2,1)$;

$\underline{\underline{I}}(1,1),\underline{\underline{I}}(2,2)$; $\underline{\underline{I}}(1,2),\underline{\underline{I}}(2,1)$; $\underline{\underline{A}}(1,1),\underline{\underline{O}}(1,1)$; $\underline{\underline{A}}(2,1),\underline{\underline{O}}(1,2)$

$\underline{\underline{O}}(1,2),\underline{\underline{A}}(1,1)$; $\underline{\underline{O}}(2,2),\underline{\underline{A}}(1,1)$; $\underline{\underline{O}}(3,4),\underline{\underline{A}}(1,2)$; $\underline{\underline{O}}(3,3),\underline{\underline{A}}(2,1)$

Comments on the output of Problem 4A

Some typical examples of the outputs obtained in the above tests are given below in Table 3. In all the outputs, wherever G1 or G2 is F, showing disagreement between the l.h.s and r.h.s of Eq.(1) or (2), a ring has been put round this entry. It will be noticed that the rings are very few and far between, but it would be highly desirable to obtain definite rules for identifying their occurrence. Therefore.

A(1.1)		A(1,2)					
A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	F	F	T
T	F	T	T	T	F	F	T
T	D	D	T	Ⓕ	D	F	Ⓕ
T	X	X	X	T	X	X	T
F	T	F	F	T	F	F	T
F	F	F	F	T	F	F	T
F	D	F	F	T	F	F	T
F	X	X	X	T	X	X	T
D	T	D	D	T	F	F	T
D	F	D	D	T	F	F	T
D	D	D	D	T	D	F	Ⓕ
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T
		DISAGREE			DISAGREE		

$$|P| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$$

$$|R'| = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Table 3(a)

A(1.1) . A(2.2)

A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	F	F	T
T	F	F	F	T	F	F	T
T	D	D	D	T	F	F	T
T	X	X	X	T	X	X	T
F	T	F	F	T	F	F	T
F	F	T	T	T	F	F	T
F	D	D	D	T	F	F	T
F	X	X	X	T	X	X	T
D	T	D	D	T	F	F	T
D	F	D	D	T	F	F	T
D	D	D	D	T	D	F	Ⓕ
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T
			AGREE			DISAGREE	

$$\underline{\underline{R}} = \underline{\underline{E}}(1, 1)$$

$$|P| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = |E(1, 1)|$$

$$|R'| = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Table 3(b)

A(1,1) . A(1,3)

A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	T	T	T
T	F	T	T	T	F	F	T
T	D	T	T	T	D	D	T
T	X	X	X	T	X	X	T
F	T	F	F	T	F	F	T
F	F	F	F	T	F	F	T
F	D	F	F	T	F	F	T
F	X	X	X	T	X	X	T
D	T	D	D	T	D	D	T
D	F	D	D	T	F	F	T
D	D	D	D	T	D	D	T
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T

AGREE AGREE

$\underline{\underline{P}} \subset \underline{\underline{Q}}$

Table 3(c)

$$|P| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$|R'| = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$$

C(2.1) . O(1,2)

A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	T	T	T
T	F	T	T	T	F	F	T
T	D	T	T	T	D	D	T
T	X	X	X	T	X	X	T
F	T	T	T	T	F	F	T
F	F	T	T	T	T	T	T
F	D	T	T	T	D	D	T
F	X	X	X	T	X	X	T
D	T	T	T	T	D	D	T
D	F	T	T	T	D	D	T
D	D	D	T	F	D	D	T
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T
			DISAGREE			AGREE	

$$\underline{\underline{R}}' = \underline{\underline{E}}(1,\ 1)$$

Table 3(d)

$$|P| = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$|R'| = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = E(1,\ 1)$$

C(1.1) . O(1.2)

A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	T	T	T
T	F	T	T	T	T	T	T
T	D	T	T	T	T	T	T
T	X	X	X	T	X	X	T
F	T	T	T	T	F	F	T
F	F	T	T	T	F	F	T
F	D	D	T	Ⓕ	D	F	Ⓕ
F	X	X	X	T	X	X	T
D	T	T	T	T	D	D	T
D	F	T	T	T	D	D	T
D	D	D	T	Ⓕ	D	D	T
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T

$$|P| = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$|R'| = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$$

Table 3(e)

E(1.3) . E(3.2)

A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	F	F	T
T	F	T	T	T	T	T	T
T	D	T	D	T	D	D	T
T	X	X	X	T	X	X	T
F	T	F	F	T	F	F	T
F	F	T	T	T	F	F	T
F	D	D	D	T	F	F	T
F	X	X	X	T	X	X	T
D	T	D	D	T	F	F	T
D	F	T	T	T	D	D	T
D	D	D	D	T	D	D	T
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T

AGREE   AGREE

$$|P| = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$|R'| = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$\underline{\underline{P}}^{c} = \underline{\underline{Q}}$ P and Q have two 1's in a row or column

Table 3(f)

I(1.2) . I(2.1)

A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	F	F	T
T	F	T	T	T	T	T	T
T	D	T	T	T	D	D	T
T	X	X	X	T	X	X	T
F	T	T	T	T	T	T	T
F	F	T	T	T	F	F	T
F	D	T	T	T	D	D	T
F	X	X	X	T	X	X	T
D	T	T	T	T	D	D	T
D	F	T	T	T	D	D	T
D	D	D	T	Ⓕ	D	D	T
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T

$$|P| = \begin{pmatrix} 0 & \cdot 1 \\ 1 & 1 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$|R'| = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} =$$

AGREE

$$\underline{\underline{R}}' = \underline{\underline{E}}(1, 2)$$

Table 3(g)

C(1.2) . A(1,1)

A	B	R1	R2	G1	R3	R4	G2
T	T	T	T	T	T	T	T
T	F	T	T	T	F	F	T
T	D	T	T	T	D	D	T
T	X	X	X	T	X	X	T
F	T	F	F	T	F	F	T
F	F	T	T	T	F	F	T
F	D	D	D	T	F	F	T
F	X	X	X	T	X	X	T
D	T	D	D	T	D	D	T
D	F	T	T	T	F	F	T
D	D	D	D	T	D	D	T
D	X	X	X	T	X	X	T
X	T	X	X	T	X	X	T
X	F	X	X	T	X	X	T
X	D	X	X	T	X	X	T
X	X	X	X	T	X	X	T
			AGREE			AGREE	

$$|P| = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$|Q| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$|R| = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$|R'| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$\underline{\underline{P}} \supset \underline{\underline{Q}}$$

Table 3(h)

we shall first focus attention on those cases where there is complete agreement between the left- and right-hand sides of Eq.(1), or (2). These are marked by the entry "Agree" below the corresponding table. On looking at these, it is found that the agreements occur only if one of the following three conditions occur :

(a) The condition $|\underline{\underline{P}}| \supseteq |\underline{\underline{Q}}|$, $|\underline{\underline{P}}| \subseteq |\underline{\underline{Q}}|$, is satisfied, when both sum and product agree.

(b) The condition $|\underline{\underline{P}}| = |\underline{\underline{Q}}^{c}|$ is satisfied, and $|\underline{\underline{P}}|$ and $|\underline{\underline{Q}}|$ have two 1's when also both sum and product agree.

(c) The matrix of the resultant operator $\underline{\underline{R}}$, or $\underline{\underline{R}}'$, is $|\underline{\underline{E}}|$ or $|\underline{\underline{N}}|$, in which case either sum or product agrees, whichever leads to the "equivalence" operator $\underline{\underline{E}}(k,\ell)$, $k, \ell = 1,2$.

Very similar conditions are found to be true also in QLOGIC and still more generally in GLOGIC, for $\underset{\approx}{Z}(\underset{\approx}{k}, \underset{\approx}{\ell})$, and a theoretical justification of these is given for the general case in Part 3.

Thus, Eqs.(1) and (2) are not valid for SNS truth values as a general rule, but only under the restricted conditions given in (a), (b), (c) above. However, as will be shown in Part 3, for "Boolean truth values" defined by the function BTV(SVA,SMZ,SVB)$\Longleftrightarrow$SBINPT(SVA,SMZ,SVB) = SVC, which corresponds to the Boolean algebraic equation $\langle \underline{\underline{a}}|\underline{\underline{Z}}|\underline{\underline{b}}\rangle = c =$ BTV, the analogue of Eq.(1) using the Boolean sum is always true, while that of Eq.(2) employing the Boolean product is not necessarily true.

However, for the standard logical connectives $\underline{\underline{E}}(= \underline{\underline{E}}(1, 1))$ and $\underline{\underline{N}} = (= \underline{\underline{E}}(1, 2))$, which can be defined both as conjunction and disjunction of relations involving $\underline{\underline{A}}(k, \ell)$ and $\underline{\underline{O}}(k, \ell)$, <u>both</u> Eqs.(1) and (2) are valid. Thus, three different, but equivalent, definitions of $\underline{\underline{E}}(1, 1)$ can be given, as in (3a),(3b) and (3c) below:

$$\underline{\underline{E}}(1, 1) = |s(1)\rangle \Longleftrightarrow \langle s(1)| \qquad (3a)$$

$$= \underline{\underline{A}}(1,1) \oplus \underline{\underline{A}}(2,2) \qquad (3b)$$

$$= O(2,1) \otimes O(1,2) \qquad (3c)$$

All of them lead to the same matrix as shown in 4(a,b,c).

$$|E| = \begin{pmatrix}1\\0\end{pmatrix} \equiv (1 \quad 0) \qquad = \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix} \qquad (4a)$$

$$\begin{pmatrix}1 & 0\\0 & 0\end{pmatrix} \oplus \begin{pmatrix}0 & 0\\0 & 1\end{pmatrix} = \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix} \qquad (4b)$$

$$\begin{pmatrix}1 & 0\\1 & 1\end{pmatrix} \otimes \begin{pmatrix}1 & 1\\0 & 1\end{pmatrix} = \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix} \qquad (4c)$$

In fact, the general definition of $\underline{\underline{E}}(k, \ell)$ as SME,SK,SL holds in a manner very similar to Eqs(4a,b,c), as was verified by detailed computer checking <u>via</u> MATLOG using Eqs.(5a,b,c) below and the subroutine SSTV employing the BVMF binary product $(\underline{\underline{a}}|\underline{\underline{Z}}|\underline{\underline{b}}) = \underline{\underline{c}}$. An algebraic checking of this, following the pattern in Eqs.(4a,b,c), is given in Table 4 below in the section dealing with comments on Problem 4B.

This means that the ten standard connectives, $\underline{\underline{A}}(k,\ell)$, $\underline{\underline{O}}(k,\ell)$,a $\underline{\underline{E}}(k,\ell)$ (namely $\underline{\underline{E}}(1, 1)$ and $\underline{\underline{E}}(1, 2)$) can be treated freely using the SSTV formalism without any difficulty.

Probmem 4B ~~In ord~~er to check this for all $\underline{E}(k, \ell)$ with $k, \ell =$ 1 to 4, the results given by three ways of defining the equivalence operator as given in Eq.(5) below, can be tested <u>via</u> the MATLOG statements following them.

$$\underline{\underline{E}}(k, \ell) = \big|s(k)\big\rangle \Longleftrightarrow \big\langle s(\ell)\big| \qquad (5\,a)$$

$$\underline{\underline{E}}(k, \ell) = \underline{\underline{A}}(k, \ell) \oplus \underline{\underline{A}}(k^c, \ell^c) \qquad (5b)$$

$$\underline{E}(k, \ell) = \underline{\underline{O}}(k^c, \ell) \otimes \underline{\underline{O}}(k , \ell^c) \qquad (5c)$$

```
   SVC1 =  SSTV(SVA,SME,SK,SL,SVB)  <-->  (4a)

   SVC2 =  SSTV(SVA1,SMO,S1,S1,SVA2) <--> (4b)
where
   SVA1 =  SSTV(SVA,SMA,SK,SL,SVB)
   SVA2 =  SSTV(SVA,SMA,SCOMP(SK),SCOMP(SL),SVB)

   SVC3 =  SSTV(SVI1,SMA,S1,S1,SVI2)  <-->  (4c)
where
   SVI1 =  SSTV(SVA,SMO,SCOMP(SK),SL,SVB)
   SVI2 =  SSTV(SVA,SMO,SK,SCOMP(SL),SVB)
```

Make the check for all $\underline{E}(k, \ell)$, $k, \ell =$ 1 to 4 and print them. Verify that the three values SVC1, SVC2 and SVC3 agree in each case.

Comments on Problem 4B

It is found that the three different ways of defining $\underline{\underline{E}}(k, \ell)$, as in (5a), (5b) and (5c), lead to identically the same truth values for SVC1, SVC2 and SVC3, for all k, ℓ, and for all combinations of SVA and SVB. This can be seen even more clearly, from the matrix formalism viewpoint , in Table 4, where the extension of the definition of $\underline{\underline{E}}(k, \ell)$, from $\underline{\underline{E}}(1, 1)$ and $\underline{\underline{E}}(1, 2)$, to all possible combinations of k, ℓ is illustrated. The three ways of defining $\underline{\underline{E}}$ are given respectively in columns 3, 2 and 4. It is very interesting to note that, formally, all the possible matrices of the type $\underline{\underline{E}}(k, \ell)$ cover the eight 2x2 matrices that are left, after considering $\underline{\underline{A}}(k, \ell)$ and $\underline{\underline{O}}(k, \ell)$ for $k, \ell = 1, 2$. It is also reassuring to note that the formal Boolean algebraic definition of the logical operators $\underline{\underline{A}}$, $\underline{\underline{O}}$ and $\underline{\underline{E}}$ as Boolean direct product, direct sum, and direct equivalence, of SK and SL is completely valid in the extended BA-2 algebra.

Problem 4C

In a similar manner, the problem in 4B can be checked using SSTV2 formalism. However, in this case, Eq.(5a) has no description via SSTV2, and therefore only the possibilities (5b) and (5c) need be tested. This can be done by using analogous MATLOG statements as those given for Problem 4B, but replacing

Table 4. Test of the equivalence of different definitions of $\underline{\underline{E}}(k, \ell)$ in the extended BA-2 algebra, employing truth values T, F, D, X†

k, ℓ	$\underline{\underline{A}}(k, \ell) \oplus \underline{\underline{A}}(k^c, \ell^c)$	$\underline{\underline{E}}(k, \ell)$	$\underline{\underline{Q}}(k^c, \ell) \otimes \underline{\underline{Q}}(k, \ell^c)$
(1, 1)	$\begin{pmatrix}1 & 0\\0 & 0\end{pmatrix} \oplus \begin{pmatrix}0 & 0\\0 & 1\end{pmatrix}$	$\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}$	$\begin{pmatrix}1 & 0\\1 & 1\end{pmatrix} \otimes \begin{pmatrix}1 & 1\\0 & 1\end{pmatrix}$
(1, 2)	$\begin{pmatrix}0 & 1\\0 & 0\end{pmatrix} \oplus \begin{pmatrix}0 & 0\\1 & 0\end{pmatrix}$	$\begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}$	$\begin{pmatrix}0 & 1\\1 & 1\end{pmatrix} \otimes \begin{pmatrix}1 & 1\\1 & 0\end{pmatrix}$
(1, 3)	$\begin{pmatrix}1 & 1\\0 & 0\end{pmatrix} \oplus \begin{pmatrix}0 & 0\\0 & 0\end{pmatrix}$	$\begin{pmatrix}1 & 1\\0 & 0\end{pmatrix}^*$	$\begin{pmatrix}1 & 1\\1 & 1\end{pmatrix} \otimes \begin{pmatrix}1 & 1\\0 & 0\end{pmatrix}$
(1, 4)	$\begin{pmatrix}0 & 0\\0 & 0\end{pmatrix} \oplus \begin{pmatrix}0 & 0\\1 & 1\end{pmatrix}$	$\begin{pmatrix}0 & 0\\1 & 1\end{pmatrix}^*$	$\begin{pmatrix}0 & 0\\1 & 1\end{pmatrix} \otimes \begin{pmatrix}1 & 1\\1 & 1\end{pmatrix}$
(3, 3)	$\begin{pmatrix}1 & 1\\1 & 1\end{pmatrix} \oplus \begin{pmatrix}0 & 0\\0 & 0\end{pmatrix}$	$\begin{pmatrix}1 & 1\\1 & 1\end{pmatrix}$	$\begin{pmatrix}1 & 1\\1 & 1\end{pmatrix} \otimes \begin{pmatrix}1 & 1\\1 & 1\end{pmatrix}$
(3, 4)	$\begin{pmatrix}0 & 0\\0 & 0\end{pmatrix} \oplus \begin{pmatrix}0 & 0\\0 & 0\end{pmatrix}$	$\begin{pmatrix}0 & 0\\0 & 0\end{pmatrix}$	$\begin{pmatrix}0 & 0\\0 & 0\end{pmatrix} \otimes \begin{pmatrix}1 & 1\\1 & 1\end{pmatrix}$

*For k, ℓ equal to (3, 1) and (4, 1), the other two possible 2 x 2 matrices $\begin{pmatrix}1 & 0\\1 & 0\end{pmatrix}$ and $\begin{pmatrix}0 & 1\\0 & 1\end{pmatrix}$ will be similarly obtained for the matrix $|\underline{\underline{E}}(k, \ell)|$.

† It may be verified that the condition (a), or (c), derived from Problem 4A, is satisfied in all the cases, provided we also note that, for all $\underline{\underline{P}}$,
$\begin{pmatrix}1 & 1\\1 & 1\end{pmatrix} \supseteq \underline{\underline{P}}$ and $\underline{\underline{P}} \supseteq \begin{pmatrix}0 & 0\\0 & 0\end{pmatrix}$

symbol SSTV2 by SSTV in all places. Calling these outputs for (5b) and (5c) as SVC2P and SVC3P respectively, Problem 4C can be stated as follows.

Make the calculation for all $\underline{\underline{E}}(k, \ell)$, $k, \ell = 1$ to 4 with SSTV2 and print the results for SVC2P and SVC3P in each case. Check them for agreement.

Comments on Problem 4C

The full computer output has not been obtained yet*, but the essential features are evident from a consideration of the simple example of $\underline{\underline{E}}(1, 1)$. It is found that the 4 x 4 truth tables as given by SSTV and SSTV2 do not fully agree, and what is more, the two different definitions in Eqs. (5b) and (5c) which can be programed in SSTV2 do not themselves fully agree with one another. Table 5 below contains the full 4x4 truth tables corresponding to Eq.(5a), using SSTV, and for Eqs.(5b), (5c), using SSTV2. Although for T, F and D there is complete agreement between all three tables, the two entries for (D, X) and (X, D) are completely different in the three cases — namely, equal to X for SVC1, F for SVC2P and T for SVC3P. Thus, it appears that the extension from classical logic into the

*This has been obtained and will be discussed in the appendix to Part III, MR-62. (Added 10.9.87).

Table 5. Comparison of 4x4 truth tables for E(1, 1) calculated via the subroutine SSTV2 for SVC2P and SVC3P with that for SSTV

	SVC1* (in SSTV)				SVC2P (in SSTV2)				SVC3P (in SSTV2)			
a \ b	T	F	D	X	T	F	D	X	T	F	D	X
T	T	F	D	X	T	F	D	X	T	F	D	X
F	F	T	D	X	F	T	D	X	F	T	D	X
D	D	D	D	X	D	D	D	Ⓕ	D	D	D	Ⓣ
X	X	X	X	X	X	X	Ⓕ	X	X	X	Ⓣ	X

*SVC2 and SVC3 completely agree with the table for SVC1.

BA-2 algebra of SNSLOGIC is consistent only for the matrix formalism and leads to inconsistencies if it is extended via truth values. Although the discrepancies occur only for the extremely artificial situation of the input and output being tautology (D) and contradiction (X) respectively, still, for mathematical consistency, one should expect the algebra to give identical results. It is not yet clear* whether any logical redefinition of the functions SSTV2 could lead to complete consistency between different ways of defining the equivalence operator. This should be discussed after similar problems are considered in QLOGIC and GLOGIC*.

* They are fully explicable (See MR-62, appendix).

MATLOG Program in FORTRAN

Part II : MATLOG Program for quantifier logic and their tests

G.N. Ramachandran
INSA Albert Einstein Professor

and

T.A. Thanaraj
Visiting Fellow

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

*(Permanent Address: Centre for Cellular & Molecular Biology,
Hyderabad 500 007.)

MR - 61, June 1987

MATLOG Program in FORTRAN

Part II: MATLOG Programs for quantifier logic and their tests

G.N. Ramachandran
INSA Albert Einstein Professor

and

T.A. Thanaraj*
Visiting Fellow

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

*(Permanent Address: Centre for Cellular and Molecular Biology, Hyderabad 500 007.)

MR-61 , June 1987

CONTENTS

Page No.

1. Quantifier logic in BVMF 1

(a) General 1

(b) Boolean algebraic formulae for quantifiers in general 2

(c) Logical operators in QL-2 6

2. Problems in QL-2 algebra 11

Problems 5, 6, 7 11

Problem 8 12

3. Comments on the outputs from the problems in Section 2 13

Problem 5 13

Problem 6 16

Problem 7 18

4. QL-1 algebra (PLOGIC 22

(a) Outline of basic theory 22

(b) MATLOG functions for QL-1 29

(c) Theory of the QL-1A binary relation 34

(d) Theory of the QL-1A unary relation 36

(e) Illustrations and problems in PLOGIC 43

Problems 9A and B and Comments 43

Problems 10A and B 44

Comments on the output of Problems 10A,B 47

Problem 11A,B 50

Solution of the Problems 11A,B 50

Problem 12A,B 53

Solution to Problem 12A 53

Solution to Problem 12B 55

(f) Complementation of the name of an SNS logical connective 56

(g) Construction of the matrix of a "logical" connective in QL-2 57

CONTENTS

Page No.

5. Multiple quantifiers and their implementation 61

(a) Elementary statement in QL-1B 61

(b) Implementation of QL-1B statements in predicate logic *via* PQLOG 63

(i) QL-1B binary relation 63

(ii) QL-1B unary relation 65

(c) Subroutines in PQLOG 65

Part II: MATLOG Programs for quantifier logic and their tests

1. Quantifier logic in BVMF

(a) General

The theory of quantifiers in BVMF has been developed first in MR-25A which has been published in Current Science (Parts I and II , Vol 52, pages 292 and 335, 1983). Since then, it has undergone several revisions, e.g. in MR-33, 45 46A, 48, 49, 50 and 51. However, a definitive treatment of the theory of quantifiers *via* well-formed formulae in SNSLOG was given in MR-52, and this laid the basis for the distinction between QL-1 and QL-2 as two standard types of relations in the BVMF description of quantified statements. The essential differences between these two are indicated by the following equations.

$$\text{QL-1} \;:\; (\underset{\sim}{q_Z}x)(\underline{\underline{a}}x\ \underline{\underline{Z}}(k,\ell)\ \underline{\underline{b}}x)\ ,\quad \text{e.g. } (\exists x)(\underline{\underline{a}}x \Longrightarrow \underline{\underline{b}}x)$$

$$\text{QL-2} \;:\; \underset{\sim}{a}x\ \underset{\sim}{Z}(k,\ell)\ \underset{\sim}{b}y\ ,\quad \text{e.g. } (\forall x)(\underline{\underline{a}}x) \Longrightarrow (\exists y)(\underline{\underline{b}}y)$$

The treatment of these two types in BVMF algebra are clearly different. The latter, which is first considered, can be implemented in exactly the same manner as SNS, but using 3-vectors and 3x3 Boolean matrices. For doing this, a set of three basic states (1 0 0),(0 1 0), (0 0 1), denoted by "For all", "For some", "For none", are used, and these lead to the eight possible quantifier states Q1 to Q8

which are described in MR-52. The implementation in QL-2 is particularly fully treated in MR-53, and the implementation in QL-1 in MR-54. These are reviewed and summarized in MR-56, Lectures 3 and 4, Series 2.

In our treatment below, we shall designate QL-2 as QLOGIC and QL-1 as PLOGIC. In the case of QL-2, the algebraic subroutines are identical with those in SNS (i) to (xvi) and they are briefly noted here, but the logical subroutines are described. in fuller detail.

(b) Boolean algebraic formulae for quantifiers in general

These formulae are equally valid for QL-1 and QL-2, and are the basis for the implementation of quantifier states in EVMF. As already mentioned, they utilize Boolean 3-vectors and Boolean 3x3 matrices. The basic 16 subroutines in SNSALG can be taken over practically unchanged for the QLOGIC also. The names of these are listed below, but only brief details are given, since they could be obtained by replacing the names of vectors, matrices, and functions, of the form SXXXXX, by QXXXXX, and remembering that M = N = 2 for the former get changed to M = N = 3 for the latter. The symbols adopted for programming QLOGIC are given in Table 1.

Table 1. Description of quantifier states in the EVMF program MATLOG

Description	Name printed	Symbol in program	Description in BA-3	Symbol in QBA	Description in standard logic
For all	ALL	Q1	(1 0 0)	$\forall$	$(\forall x)(\underline{\underline{g}}x)$
Not for all	NAL	Q2	(0 1 1)	Λ	$\neg(\forall x)(\underline{\underline{g}}x)$
For some	SOM	Q3	(0 1 0)	Σ	$(\exists x)(\underline{\underline{g}}x) \wedge (\exists x)(\neg \underline{\underline{g}}x$
All or none	AON	Q4	(1 0 1)	Θ	$(\forall x)(\underline{\underline{g}}x) \vee (\forall x)(\neg \underline{\underline{g}}x$
Not exists	NEX	Q5	(0 0 1)	Φ	$\neg(\exists x)(\underline{\underline{g}}x)$
There exists	EXS	Q6	(1 1 0)	$\exists$	$(\exists x)(\underline{\underline{g}}x)$
Indefinite	IND	Q7	(1 1 1)	Δ	$(\exists x)(\underline{\underline{g}}x) \vee \neg(\exists x)(\underline{\underline{g}}x)$
Impossible or Contradictory	XXX	Q8	(0 0 0)	ϕ	$(\exists x)(\underline{\underline{g}}x) \wedge \neg(\exists x)(\underline{\underline{g}}x)$

The matrix connectives QMZ(I,J) are 3x3 matrices, with I,J = 1, 2, 3 and the quantified terms are 3-vectors QVA(I), I = 1, 2, 3 which can be any one of the eight BA-3 states Q1 to Q8. The states Q1, Q3, Q5 form the generators of the BA-3 algebra and are called "basic states". The logical connectives $\underset{\sim}{A}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{O}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{I}(\underset{\sim}{k}, \underset{\sim}{\ell})$ are defined in a manner closely similar to the SNS $\underline{\underline{z}}(k, \ell)$ -- e.g.

QMA,QK,QL = QMAT(QK , QL) (See (xi) below)

The subroutines for the BA-3 operators (i) to (xvi) are as follows.

(i) Boolean sum of two vectors $\underset{\sim}{a} \oplus \underset{\sim}{b} = \underset{\sim}{c}$ $(\oplus = \underset{\sim}{U} = \text{union})$

QUNION (QVA,QVA) = QVC

BSUM (QVA(1),QVB(1)) = QVC(1)

BSUM (QVA(2),QVB(2)) = QVC(2)

BSUM (QVA(3),QVB(3)) = QVC(3)

(ii) Boolean product of two vectors $\underset{\sim}{a} \otimes \underset{\sim}{b} = \underset{\sim}{c}$ $(\otimes = \underset{\sim}{V} = \text{vidya})$

QVIDYA(QVA,QVB) = QVC

BPDT (QVA(1),QVB(1)) = QVC(1)

BPDT (QVA(2),QVB(2)) = QVC(2)

BPDT (QVA(3),QVB(3)) = QVC(3)

(iii) Boolean complement of Q-vector $\underset{\sim}{a}^c = \underset{\sim}{b}$

QCOMP(QVA) = QVB

BCMP (QVA(1)) = QVB(1)

BCMP (QVA(2)) = QVB(2)

BCMP (QVA(3)) = QVB(3)

(iv) Boolean sum of two matrices $\underset{\sim}{P} \oplus \underset{\sim}{Q} = \underset{\sim}{R}$

QMXSUM(QMP,QMQ) = QMR

BSUM(QMP(I,J), QMQ(I,J)) = QMR(I,J), I, J = 1, 2, 3

(v) Boolean product of two matrices $\underset{\sim}{P} \otimes \underset{\sim}{Q} = \underset{\sim}{R}$

QMXPDT(QMP,QMQ) = QMR

Replace BSUM in (iv) by BPDT .

(vi) Boolean complement of a matrix operator $\underset{\sim}{P}^c = \underset{\sim}{Q}$

QMCMP(QMP) = QMQ

BCMP(QMP(I,J)) = QMQ(I,J), I = 1, 2, 3

(vii) Scalar product of two vectors

$$\langle \underset{\sim}{a} \mid \underset{\sim}{b} \rangle = \underset{\sim}{c} = a_1 \otimes b_1 \oplus a_2 \otimes b_2 \oplus a_3 \otimes b_3$$

QSCPDT(QVA,QVB) = BC (Note Boolean scalar on the r.h.s)

(viii) Unary product of a vector with a matrix

$$\langle \underset{\sim}{a} | \underset{\sim}{Z} | = \langle \underset{\sim}{b} | , \quad \bigoplus_i a_i \otimes Z_{ij} = b_j , \quad i,j = 1,2,3$$

QUNPDT(QVA,QMZ) = QVB

(ix) Binary product for the matrix with two vectors

$$\langle \underset{\sim}{a} | \underset{\sim}{Z} | \underset{\sim}{b} \rangle = c \; ; \quad \bigoplus_i \bigoplus_j a_i \otimes Z_{ij} \otimes b_j = c , \quad i,j = 1,2,3$$

QBINPT (QVA,QMZ,QVB) = BC (Note Boolean scalar on the r.h.s)

(x) Matrix product of two matrices $\bigoplus_j P_{ij} \otimes Q_{jk} = R_{ik}$, $i,j,k = 1, 2, 3$

QMATPT(QMP,QMQ) = QMR

(xi) Direct product of two vectors $\underset{\sim}{a} \times \underset{\sim}{b} = \underset{\sim}{Z}$

QDIRPT(QVA,QVB) = QMZ

QMZ(I,J) = BPDT(QVA(I),QVB(J))

(xii) Direct sum of two vectors $\underset{\sim}{a} + \underset{\sim}{b} = \underset{\sim}{Z}$

QDIRSM(QVA,QVB) = QMZ

QMZ(I,J) = BSUM(QVA(I),QVB(J))

(xiii) Direct equivalence of two vectors $(\underset{\sim}{a} \equiv \underset{\sim}{b}) = \underset{\sim}{Z}$

QDIREQ(QVA,QVB) = QMZ

QMZ(I,J) = BEQU(QVA(I),QVB(J))

(xiv) Scalar-vector direct product $a \times \underset{\sim}{b} = \underset{\sim}{c}$

QSVDPT(BA,QVB) = QVC (Note BA on the l.h s)

QVC(I) = BPROD(BA,QVB(I))

(xv) Scalar vector direct sum $a + \underset{\sim}{b} = \underset{\sim}{c}$

QSVDSM(BA,QVB) = QVC (Note BA on the l.h.s)

QVC(I) = BSUM(BA,QVB(I))

(xvi) Transpose of a matrix $\underset{\sim}{P}^t = \underset{\sim}{Q}$

QTRANS(QMP) = QMQ

QMQ(I,J) = QMP(J,I), I,J = 1, 2

(c) Logical operators in QL-2

Similar to the SNS logical functions described in Section 2(b) of Part I, the corresponding QL-2 functions, using Q1 to Q8 for the constants, can be defined. These are as follows.

(xvii) Matrix elements of logical connectives

$\underset{\sim}{A}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{O}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{I}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{k}, \underset{\sim}{\ell}$ = 1 to 8

QMATEL(QMZ,QK,QL) = QMM(I,J)

Replace S by Q, and use QK,QL with K,L = 1 to 8 in (xvii) of SNSLOG.

For QMA,QK,QL, QMM = QDIRPT(QK,QL)

For QMO,QK,QL, QMM = QDIRSM(QK,QL)

For QME,QK,QL, QMM = QDIREQ(QK,QL)

For QMI,QK,QL, QMM = QDIRSM(QCOMP(QK),QL)

(xviii) . <u>Relative truth value of one quantified term for another</u> : $\underline{\underline{t}}(\underset{\sim}{a} \mid \underset{\sim}{b}) = (\langle a \mid b \rangle, \langle a \mid b^c \rangle) = (c_1, c_2) = \underline{\underline{c}}$

QRELTV(QVA,QVB) = SVC   (Note SNS in r.h.s)

SVC(1) = QSCPDT (QVA,QVB)

SVC(2) = QSCPDT (QVA,QCOMP(QVB))

Input is in QL-2, output is in SNS

(xix) . <u>SNS truth value of a binary relation for given inputs</u> $\underset{\sim}{a}, \underset{\sim}{b}$: $\underline{\underline{t}}(\underset{\sim}{a}\ \underset{\sim}{Z}\ \underset{\sim}{b}) = \underline{\underline{c}} = (c_1, c_2)$

$c_1 = \langle \underset{\sim}{a} | \underset{\sim}{Z} | \underset{\sim}{b} \rangle, \quad c_2 = \langle \underset{\sim}{a} \mid \underset{\sim}{Z}^c \mid \underset{\sim}{b} \rangle$

QSTV (QVA,QMZ,QVB) = SVC   (Note SNS in r.h.s)

SVC(1) = QBINPT(QVA,QMZ,QVB)

SVC(2) = QBINPT(QVA, QMXCMP(QMZ),QVB)

(xx) . <u>SNS truth value of a relation involving the QL-2 logical connective</u> $\underset{\sim}{Z}(k, \ell)$ <u>for inputs</u> $\underset{\sim}{a}$, $\underset{\sim}{b}$

$\underline{\underline{t}}(\underset{\sim}{a}\ \underset{\sim}{Z}\ \underset{\sim}{b}) = \underline{\underline{c}}$ for $\underset{\sim}{Z} = \underset{\sim}{Z}(k, \ell)$ <u>via</u> relative truth values

QSTV2(QVA,QMZ,QVB) = SVC   (Note SNS in r.h.s)

QMZ = QMA,QK,QL etc. (for $\underset{\sim}{Z} = \underset{\sim}{A}, \underset{\sim}{O}, \underset{\sim}{I}, \underset{\sim}{E}$)

If QMZ = QMA,QK,QL

SVC(1) = BPDT(QSCPDT(QVA,QK), QSCPDT(QVB,QL))

SVC(2) = BSUM(QSCPDT(QVA,QCOMP(QK)),
QSCPDT(QVB,QCOMP(QL)))

If QMZ = QMO,QK,QL

SVC(1) = BSUM(QSCPDT(QVA,SK),QSCPDT(QVB,QL))

SVC(2) = BPDT(QSCPDT(QVA,QCOMP(SK)),QSCPDT(QVB,QCOMP(QL)))

```
  IF  QMZ = QME,QK,QL

      QKP = QCOMP(QK), QLP = QCOMP(QL)

     SVCA = QSTV2(QVA,QMA,QK,QL, QVB)
     SVCB = QSTV2(QVA,QMA,QKP,QLP, QVB)
     SVC(1)  = BSUM(QVCA(1),QVCB(1))
     SVC(2)  = BPDT(QVCA(2),QVCB(2))

 IF  QMZ = QMI,QK,QL
     QKP = QCOMP(QK)
     QMZ = QMO,QKP,QL
```

As mentioned for SNS logical connectives, in the case of QL-2 also, general 3x3 matrices can also occur, for relations in QL-2 that cannot be expressed in one of the four forms described above.

(xxi) Unary relation for $\underset{\sim}{Z}(\underset{\sim}{k}, \underset{\sim}{\ell})$ via relative truth value $\langle a \mid q(k)\rangle\ Z\ \langle q(\ell) \mid\ =\ b\ ,\ \ Z = \otimes, \oplus\ ,$ etc.

```
      QUNPT2(QVA,QMZ) = QVB

 IF   QMZ = QMA,QK,QL
       BAP = QSCPDT(QVA,QK)
       QVB = QSVPDT(BAP,QL)

 IF   QMZ = QMO,QK,QL
       BAP = QSCPDT(QVA,QK)
       QVB = QSVDSM(BAP,QL)
```

```
IF   QMZ = QMI,QK,QL

          BAP = QSCPDT(QVA,QCOMP(QK))
          QVB = QSVDSM(BAP,QL)

IF   QMZ = QME,QK,QL

          QKP = QCOMP(QK),   QLP = QCOMP(QL)
         QVBA = QUNPT2(QVA, QMA,QK,QL)
         QVBB = QUNPT2(QVA, QMA,QKP,QLP)
          QVB = QUNION(QVBA,QVBB)
```

Here also, the formulae closely follow those in SNS except that S is to be replaced by Q,leaving B unchanged, in SUNPT2.

(xxii) Binary matrix-Boolean operator G "agree" for checking the equivalence (agreement) of two quantifier terms $\underset{\sim}{a}$ and $\underset{\sim}{b}$: $(\underset{\sim}{a})\underset{\sim}{G}|\underset{\sim}{b}) = \underset{\sim}{c}$

```
      QMBG(QVA,QVB) = QVC        (Note SNS in r.h.s)

      BCA = BEQU(QVA(1),QVB(1))
      BCB = BEQU(QVA(2),QVB(2))
      BCC = BEQU(QVA(3),QVB(3))
     BCBP = BPDT(BCA,BCB)
   SVC(1) = BPDT(BCBP,BCC)
   SVC(2) = BCMP(QVC(1))
```

(xxiii) Matrix-Boolean unary operators $\underset{\sim}{E}$, $\underset{\sim}{N}$, $\underset{\sim}{M}$, $\underset{\sim}{L}$

In this case also, replace S by Q, and 1, 2, by 1, 2, 3. The equations are

QEQU(QVA) = QVB

QVB(1) = QVA(1), QVB(2) = QVA(2) , QVB(3) = QVA(3)

QNOT(QBA) = QVB

QVB(1) = QVA(3), QVB(2) = QVA(2) , QVB(3) = QVA(1)

QCOMP(QVA) = QVB

QVB(1) = BCMP(QVA(1)), QVB(2) = BCMP(QVA(2)),
QVB(3) = BCMP(QVA(3)).

QELL(QVA) = QVB

QVB(1) = BCMP(QVA(3)), QVB(2) = BCMP(QVA(2)),
QVB(3) = BCMP(QVA(1)).

The above subroutines appear to be sufficient for dealing with all problems containing quantifiers which involve QL-2 algebra. However, for QL-1, quite different subroutines are needed and are treated in Section 4 under PLOGIC.

2. Problems in QL-2 algebra

Problem 5

Make direct checks of the algorithms Q(i) to Q(xvi) for agreement with ideas discussed in MR-53 and 56, Lecture 3.

Problem 6

Check the agreement between QUNPDT and QUNPT2 for the cases listed in Problem 7, and print them side by side for inputs QVA = Q1 to Q8. Verify that there is agreement for Q1 to Q7 as input and note examples where they disagree for Q8 as input.

Problem 7

Check the 8x8 truth tables for (1) Q(xix) -- QSTV and (2) Q(xx) -- QSTV2 for the following connectives.

A(1,1), A(1,6), A(2,5), A(2,6), A(5,6), A(2,7), A(1,3), A(2,4)
O(1,1), O(1,6), O(2,5), O(2,6), O(5,6), O(2,7), O(1,3), O(2,4)
I(1,1), I(1,6), I(2,5), I(2,6), I(5,6), I(2,7), I(1,3), I(2,4)
E(1,1), E(1,6), E(2,5), E(2,6), E(5,6), E(2,7), E(1,3), E(2,4)

Check that they agree for the 7x7 sub-table involving Q1 to Q7 and if so, print the output for QSTV2 and note whether they disagree for Q8 as input a or b .

2. Problems in QL-2 algebra

Problem 5

Make direct checks of the algorithms Q(i) to Q(xvi) for agreement with ideas discussed in MR-53 and 56, Lecture 3.

Problem 6

Check the agreement between QUNPDT and QUNPT2 for the cases listed in Problem 7, and print them side by side for inputs QVA = Q1 to Q8. Verify that there is agreement for Q1 to Q7 as input and note examples where they disagree for Q8 as input.

Problem 7

Check the 8x8 truth tables for (1) Q(xix)— QSTV and (2) Q(xx) — QSTV2 for the following connectives.

$$\underset{\sim}{A}(\underset{\sim}{1},\underset{\sim}{1}),\ \underset{\sim}{A}(\underset{\sim}{1},\underset{\sim}{6}),\ \underset{\sim}{A}(\underset{\sim}{2},\underset{\sim}{5}),\ \underset{\sim}{A}(\underset{\sim}{2},\underset{\sim}{6}),\ \underset{\sim}{A}(\underset{\sim}{5},\underset{\sim}{6}),\ \underset{\sim}{A}(\underset{\sim}{2},\underset{\sim}{7}),\underset{\sim}{A}(\underset{\sim}{1},\underset{\sim}{3}),\ \underset{\sim}{A}(\underset{\sim}{2},\underset{\sim}{4})$$

$$\underset{\sim}{O}(\underset{\sim}{1},\underset{\sim}{1}),\ \underset{\sim}{O}(\underset{\sim}{1},\underset{\sim}{6}),\ \underset{\sim}{O}(\underset{\sim}{2},\underset{\sim}{5}),\ \underset{\sim}{O}(\underset{\sim}{2},\underset{\sim}{6}),\ \underset{\sim}{O}(\underset{\sim}{5},\underset{\sim}{6}),\ \underset{\sim}{O}(\underset{\sim}{2},\underset{\sim}{7}),\underset{\sim}{O}(\underset{\sim}{1},\underset{\sim}{3}),\ \underset{\sim}{O}(\underset{\sim}{2},\underset{\sim}{4})$$

$$\underset{\sim}{I}(\underset{\sim}{1},\underset{\sim}{1}),\ \underset{\sim}{I}(\underset{\sim}{1},\underset{\sim}{6}),\ \underset{\sim}{I}(\underset{\sim}{2},\underset{\sim}{5}),\ \underset{\sim}{I}(\underset{\sim}{2},\underset{\sim}{6}),\ \underset{\sim}{I}(\underset{\sim}{5},\underset{\sim}{6}),\ \underset{\sim}{I}(\underset{\sim}{2},\underset{\sim}{7}),\underset{\sim}{I}(\underset{\sim}{1},\underset{\sim}{3}),\ \underset{\sim}{I}(\underset{\sim}{2},\underset{\sim}{4})$$

$$\underset{\sim}{E}(\underset{\sim}{1},\underset{\sim}{1}),\ \underset{\sim}{E}(\underset{\sim}{1},\underset{\sim}{6}),\ \underset{\sim}{E}(\underset{\sim}{2},\underset{\sim}{5}),\ \underset{\sim}{E}(\underset{\sim}{2},\underset{\sim}{6}),\ \underset{\sim}{E}(\underset{\sim}{5},\underset{\sim}{6}),\ \underset{\sim}{E}(\underset{\sim}{2},\underset{\sim}{7}),\underset{\sim}{E}(\underset{\sim}{1},\underset{\sim}{3}),\ \underset{\sim}{E}(\underset{\sim}{2},\underset{\sim}{4})$$

Check that they agree for the 7x7 sub-table involving Q1 to Q7 and if so, print the output for QSTV2 and note whether they disagree for Q8 as input $\underset{\sim}{a}$ or $\underset{\sim}{b}$.

Problem 8

This will closely follow the pattern in Problem 4.

Problem 8A -- The analogous case will not be worked out in QL-2, as a more general case will be considered in GLOGIC, and it will be proved that, in general, the conjunction of the Boolean truth values of two relations is not equivalent to the Boolean truth value of the conjunction of the two relations (represented by the Boolean product of the two matrices), while the corresponding theorem is true for disjunctions.

Therefore, the test is made only for the particular result that $\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})$ can be expressed both as $\underset{\sim}{A}(\underset{\sim}{k}, \underset{\sim}{\ell}) \oplus \underset{\sim}{A}(\underset{\sim}{k}^c, \underset{\sim}{\ell}^c)$ and $\underset{\sim}{I}(\underset{\sim}{k}, \underset{\sim}{\ell}) \otimes \underset{\sim}{I}(\underset{\sim}{k}^c, \underset{\sim}{\ell}^c)$, via QSTV(Problem 8B), but not by QSTV2 (Problem 8C). These are analogous to Problems 4B and 4C in SNS for SSTV and SSTV2 respectively.

Problem 8B :-- Define QMZ,QK,QL, for Z = A, O, E, I, from subroutine Q(xviii) and check the agreement of the three truth values SVV1, SVV2, SVV3 for $\underline{\underline{c}}$, obtained by the following MATLOG statements for $(\underset{\sim}{a} \,|\, \underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell}) \,|\, \underset{\sim}{b}) = \underline{\underline{c}}$, employing only the QSTV subroutine.

* Although MATLOG statements for Problems 8B and 8C were formulated in June 1987, they could be put on the computer only by the end of September 87 (in Hyderabad) and they fully supported the theoretical expectations stated below. (See also Appendix of MATLOG-3, MR-62 for a brief discussion). (Added 12.10.87

```
          SVV1  =  QSTV(QVA,QMZ,QVB)
     where
          QMZ = QDIREC(QK,QL)

          SVV2 = SSTV(SVA1,SMO,S1,S1,SVA2)
     where
          SVA1 = QSTV(QVA,QMA,QK,QL,QVB)
          SVA2 = QSTV(QVA,QMA,QCOMP(QK),QCOMP(QL), QVB)

          SVV3 = SSTV(SVI1,SMA,S1,S1,SVI2)
     where
          SVI1 = QSTV(QVA,QMI,QK,QL,QVB)
          SVI2 = QSTV(QVA,QMI,QCOMP(QK),QCOMP(QL),QVB)
```

Check SVV1, SVV2, SVV3 for agreement and if found true, print only SVV1. (The test is to be done for all $\underset{\sim}{k}, \underset{\sim}{\ell} =$ 1 to 8, for $\underset{\sim}{k} \leq \underset{\sim}{\ell}$).

Problem 8C : — Check the agreement of QSTV2(QVA,QME,QK,QL,QVB) = SVC with the two different, but equivalent, definitions of $\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})$, following Problem 4C in SNS. The relevant equations are as follows:

```
          SVC1 = SSTV2(SVA1,SMO,S1,S1,SVA2)
     where
          SVA1 = QSTV2(QVA,QMA,QK,QL,QVB)
          SVA2 = QSTV2(QVA,QMA,QCOMP(QK),QCOMP(QL),QVB)
```

```
        SVC2 = SSTV2(SVI1,SMA,S1,S1,SVI2)
where
        SVI1 = QSTV2(QVA,QMI,QK,QL,QVB)
        SVI2 = QSTV2(QVA,QMI,QCOMP(QK),QCOMP(QL), QVB)
```

Print both SVC1 and SVC2, and check if they agree for all $\underset{\sim}{a}$, $\underset{\sim}{b}$ = Q1 to Q7, and also with SVV1 of Problem 8B, and note where they disagree for $\underset{\sim}{a}$ or $\underset{\sim}{b}$ = Q8. (Test is to be repeated for all $(\underset{\sim}{k}, \underset{\sim}{\ell})$, with $\underset{\sim}{k}, \underset{\sim}{\ell}$ = 1 to 8, $\underset{\sim}{k} \leqslant \underset{\sim}{\ell}$).

3. Comments on the outputs from the problems in Section 2

The outputs for the functions given below are illustrative of the nature of these functions. Note that Q8 is printed as IMP and will be changed later to XXX.

Problem 5

It will be noticed from Table 1 that the four matrix-Boolean operators QMBE, QMBN, QMBM, QMBL permute the four standard quantifier states ALL, EXS, NAL, NEX among themselves, and similarly interchange pairwise the remaining four states (SOM, AON), (IND, IMP). Hence the collection of eight states of BA-3 employed in quantifier logic are left invariant by the operation of Boolean-matrix operators.

The operators QUNION and QVIDYA are also seen to produce as outputs only one of the eight quantifier states, for all combinations of the input states ($\underset{\sim}{a}$, $\underset{\sim}{b}$), and therefore, in QLOGIC, the algebra is complete under "Boolean" operators.

The scalar product of two Q-vectors and the SNS relative truth value of those are also listed in Table 1, for $\underset{\sim}{k}, \underset{\sim}{\ell}$ = 1 to 8. The former is a Boolean scalar 1 or 0 , and the latter a Boolean SNS vector T, F, D or X. It can be verified that the first component c_{α} of the SNS truth value $\underline{\underline{c}} = (c_{\alpha}, c_{\beta})$ for the latter is always equal to the Boolean number corresponding to QSCPDT, in all cases. This indicates the consistency of our quantifier algebra, employing BA-2 truth values in SNS, with the classical quantifier calculus, using only BA-1 truth values. The BA-2 calculus is analytical continuation, for logical truth values T, F, D, X, of the BA-1 calculus employing only T and F, and the latter always leads to the former if the states D and X are not taken as independent logical truth values, but expressible as $D = T \vee F$ and $X = T \wedge F$. (See MR-52 for a treatment of this aspect.)

Table 1. Outputs for some typical functions in QLOGIC

	QCOMP	QMBE	QMBN	QMBM	QMBL
ALL	NAL	ALL	NEX	NAL	EXS
NAL	ALL	NAL	EXS	ALL	NEX
SOM	AON	SOM	SOM	AON	AON
AON	SOM	AON	AON	SOM	SOM
NEX	EXS	NEX	ALL	EXS	NAL
EXS	NEX	EXS	NAL	NEX	ALL
IND	IMP	IND	IND	IMP	IMP
IMP	IND	IMP	IMP	IND	IND

QUNION

ALL	IND	EXS	AON	AON	EXS	IND	ALL
IND	NAL	NAL	IND	NAL	IND	IND	NAL
EXS	NAL	SOM	IND	NAL	EXS	IND	SOM
AON	IND	IND	AON	AON	IND	IND	AON
AON	NAL	NAL	AON	NEX	IND	IND	NEX
EXS	IND	EXS	IND	IND	EXS	IND	EXS
IND	IND	IND	IND	IND	IND	IND	IND
ALL	NAL	SOM	AON	NEX	EXS	IND	IMP

QVIDYA

ALL	IMP	IMP	ALL	IMP	ALL	ALL	IMP
IMP	NAL	SOM	NEX	NEX	SOM	NAL	IMP
IMP	SOM	SOM	IMP	IMP	SOM	SOM	IMP
ALL	NEX	IMP	AON	NEX	ALL	AON	IMP
IMP	NEX	IMP	NEX	NEX	IMP	NEX	IMP
ALL	SOM	SOM	ALL	IMP	EXS	EXS	IMP
ALL	NAL	SOM	AON	NEX	EXS	IND	IMP
IMP	IMP	IMP	IMP	IMP	IMP	IMP	IMP

QSCPD1

1	0	0	1	0	1	1	0
0	1	1	1	1	1	1	0
0	1	1	0	0	1	1	0
1	1	0	1	1	1	1	0
0	1	0	1	1	0	1	0
1	1	1	1	0	1	1	0
1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0

QRELTV

T	F	F	T	F	T	T	F
F	T	D	D	D	D	T	F
F	T	T	F	F	T	T	F
D	D	F	T	D	D	T	F
F	T	F	T	T	F	T	F
D	D	D	D	F	T	T	F
D	D	D	D	D	D	T	F
X	X	X	X	X	X	X	X

Problem 6

Four illustrations each for $\underset{\sim}{A}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{O}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})$, $\underset{\sim}{I}(\underset{\sim}{k}, \underset{\sim}{\ell})$ are copied from the computer outputs obtained for this problem. It was verified that QUNPDT and QUNPT2 give identical results for $\underset{\sim}{a}$ = QVA = Q1 to Q7 for all examples. For Q8 as input QVA, however, $\underset{\sim}{b}$ = QVB = Q8 = IMP for $\underset{\sim}{A}(\underset{\sim}{k}, \underset{\sim}{\ell})$ and $\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})$, while QVB = QL for $\underset{\sim}{O}(\underset{\sim}{k}, \underset{\sim}{\ell})$ and $\underset{\sim}{I}(\underset{\sim}{k}, \underset{\sim}{\ell})$, for QUNPT2. On the other hand, in the case of QUNPDT, QVB = Q8 whenever QVA = Q8. Hence only QUNPT2 results are reproduced, and, where they disagree for QUNPDT, they are marked by a ring.

Table 2. Typical examples of the output of QUNPT2 for QVA = 1 to 8

Input QVA	Output QVB for			
	$\underset{\sim}{A}(1, 1)$	$\underset{\sim}{A}(1, 6)$	$\underset{\sim}{A}(2, 5)$	$\underset{\sim}{A}(2, 7)$
ALL	ALL	EXS	IMP	IMP
NAL	IMP	IMP	NEX	IND
SOM	IMP	IMP	NEX	IND
AON	ALL	EXS	NEX	IND
NEX	IMP	IMP	NEX	IND
EXS	ALL	EXS	NEX	IND
IND	ALL	EXS	NEX	IND
IMP	IMP	IMP	IMP	IMP
	$\underset{\sim}{O}(1, 1)$	$\underset{\sim}{O}(1, 6)$	$\underset{\sim}{O}(2, 5)$	$\underset{\sim}{O}(2, 7)$
ALL	IND	IND	NEX	IND
NAL	ALL	EXS	IND	IND
SOM	ALL	EXS	IND	IND
ADN	IND	IND	IND	IND
NEX	ALL	EXS	IND	IND
EXS	IND	IND	IND	IND
IND	IND	IND	IND	IND
IMP	ALL	EXS	NEX	IND

Table 2. Contd.

Input QVA	Output QVB for			
	$\underset{\sim}{E}(1, 1)$	$\underset{\sim}{E}(1, 6)$	$\underset{\sim}{E}(2, 5)$	$\underset{\sim}{E}(2, 7)$
ALL	ALL	EXS	EXS	IMP
NAL	NAL	NEX	NEX	IND
SOM	NAL	NEX	NEX	IND
AON	IND	IND	IND	IND
NEX	NAL	NEX	NEX	IND
EXS	IND	IND	IND	IND
IND	IND	IND	IND	IND
IMP	IMP	IMP	IMP	IMP
	$\underset{\sim}{I}(1, 1)$	$\underset{\sim}{I}(1, 6)$	$\underset{\sim}{I}(2, 5)$	$\underset{\sim}{I}(2, 7)$
ALL	ALL	EXS	IND	IND
NAL	IND	IND	NEX	IND
SOM	IND	IND	NEX	IND
AON	IND	IND	IND	IND
NEX	IND	IND	NEX	IND
EXS	IND	IND	IND	IND
IND	IND	IND	IND	IND
IMP	ALL	EXS	NEX	IND

Even these disagreements follow a rule as mentioned above. Thus, they occur only for the O-type connectives QMO and QMI, and irrespective of $(\underset{\sim}{k}, \ell)$, the output for QUNPT2 is QVB = QL, for QVA = Q8 = IMP. This is similar to the behaviour in SNS algebra, as may be seen from Table 1B where for $\underline{\underline{O}}(k, \ell)$ and $\underline{\underline{I}}(k, \ell)$, SVA = S4 = X always leads to SL as output for SVB. The explanation of this will be given, from Boolean algebra, under even more general conditions of GLOGIC, in Part III, and it reduces to SNS and QLOGIC for M = N = 2, and 3 respectively.

Problem 7

In this case of binary relation also, the SNS truth values calculated using QSTV and QSTV2 agree for the 7x7 sub-table employing Q1 to Q7 for a and b. They, however, disagree for some examples if one of either a, or b, is Q8.

Also, as in SNS, the disagreeing examples have the truth value F, for QSTV2, for the connectives A(k, ℓ) and E(k, ℓ), while they have the truth value T for the connectives O(k, ℓ) and I(k, ℓ), as against X for the corresponding outputs for QSTV. This behaviour can also be proved still more generally for similar logical connectives defined in GLOGIC, and a description and proof are given in Part III dealing with GLOGIC. Print outs for QSTV2 in some typical cases are given in Table 2, and the disagreeing outputs are marked by rings.

Table 3. SNS output of QSTV2 for the 8x8 array of inputs QVA, QVB.

A(1.6)

T	D	T	D	F	T	D	X
P	F	F	F	F	F	F	Ⓕ
F	F	F	F	F	F	F	Ⓕ
D	D	D	D	F	D	D	Ⓕ
F	F	F	F	F	F	F	Ⓕ
D	D	D	D	F	D	D	Ⓕ
D	D	D	D	F	D	D	Ⓕ
X	Ⓕ	X	Ⓕ	Ⓕ	X	Ⓕ	X

A(5,6)

F	F	F	F	F	F	F	Ⓕ
D	D	D	D	F	D	D	Ⓕ
F	F	F	F	F	F	F	Ⓕ
D	D	D	D	F	D	D	Ⓕ
T	D	T	D	F	T	D	X
F	F	F	F	F	F	F	Ⓕ
D	D	D	D	F	D	D	Ⓕ
X	Ⓕ	X	Ⓕ	Ⓕ	X	Ⓕ	X

O(1,6)

T	T	T	T	T	T	T	Ⓣ
T	D	T	D	F	T	D	X
T	D	T	D	F	T	D	X
T	D	T	D	D	T	D	Ⓣ
T	D	T	D	F	T	D	X
T	D	T	D	D	T	D	Ⓣ
T	D	T	D	D	T	D	Ⓣ
Ⓣ	Ⓣ	Ⓣ	Ⓣ	X	Ⓣ	Ⓣ	X

O(2,5)

F	D	F	D	T	F	D	X
T	T	T	T	T	T	T	Ⓣ
T	T	T	T	T	T	T	Ⓣ
D	D	D	D	T	D	D	Ⓣ
T	T	T	T	T	T	T	Ⓣ
D	D	D	D	T	D	D	Ⓣ
D	D	D	D	T	D	D	Ⓣ
X	Ⓣ	X	Ⓣ	Ⓣ	X	Ⓣ	X

E(1,1)

T	F	F	D	F	D	D	X
F	T	T	D	T	D	D	X
F	T	T	D	T	D	D	X
D	D	D	D	D	D	D	Ⓕ
F	T	T	D	T	D	D	X
D	D	D	D	D	D	D	Ⓕ
D	D	D	D	D	D	D	Ⓕ
X	X	X	Ⓕ	X	Ⓕ	Ⓕ	X

E(5,6)

F	D	F	D	T	F	D	X
D	D	D	D	D	D	D	Ⓕ
F	D	F	D	T	F	D	X
D	D	D	D	D	D	D	Ⓕ
T	D	T	D	F	T	D	X
F	D	F	D	T	F	D	X
D	D	D	D	D	D	D	Ⓕ
X	Ⓕ	X	Ⓕ	X	X	Ⓕ	X

I(1,1)

T	F	F	D	F	D	D	X
T	T	T	T	T	T	T	Ⓣ
T	T	T	T	T	T	T	Ⓣ
T	D	D	D	D	D	D	Ⓣ
T	T	T	T	T	T	T	Ⓣ
T	D	D	D	D	D	D	Ⓣ
T	D	D	D	D	D	D	Ⓣ
Ⓣ	X	X	Ⓣ	X	Ⓣ	Ⓣ	X

I(1,6)

T	D	T	D	F	T	D	X
T	T	T	T	T	T	T	Ⓣ
T	T	T	T	T	T	T	Ⓣ
T	D	T	D	D	T	D	Ⓣ
T	T	T	T	T	T	T	Ⓣ
T	D	T	D	D	T	D	Ⓣ
T	D	T	D	D	T	D	Ⓣ
Ⓣ	Ⓣ	Ⓣ	Ⓣ	X	Ⓣ	Ⓣ	X

It has been possible to deduce the following general rule for the ringed entries that disagree with QSTV in Table 1.

$$\left.\begin{array}{l}\underset{\sim}{a} = \emptyset , \ \underset{\sim}{b} \cap q(\underset{\sim}{\ell}) \neq \emptyset \\ \text{or} \\ \underset{\sim}{b} = \emptyset , \ \underset{\sim}{a} \cap q(\underset{\sim}{k}) \neq \emptyset \end{array}\right\} \longmapsto (\underset{\sim}{a} \,|\, \underset{\sim}{Q}(\underset{\sim}{k}, \underset{\sim}{\ell}) \,|\, \underset{\sim}{b}) = T \tag{1}$$

$$\left.\begin{array}{l}\underset{\sim}{a} = \emptyset , \ \underset{\sim}{b} \cap q(\underset{\sim}{\ell}^{c}) \neq \emptyset \\ \text{or} \\ \underset{\sim}{b} = \emptyset , \ \underset{\sim}{a} \cap q(\underset{\sim}{k}^{c}) \neq \emptyset \end{array}\right\} \longmapsto (\underset{\sim}{a} \,|\, \underset{\sim}{A}(\underset{\sim}{k}, \underset{\sim}{\ell}) \,|\, \underset{\sim}{b}) = F \tag{2}$$

$$\left.\begin{array}{l}\underset{\sim}{a} = \emptyset , \ \underset{\sim}{b} \cap q(\underset{\sim}{\ell}) \neq \emptyset \\ \text{or} \\ \underset{\sim}{b} = \emptyset , \ \underset{\sim}{a} \cap q(\underset{\sim}{k}^{c}) \neq \emptyset \end{array}\right\} \longmapsto (\underset{\sim}{a} \,|\, \underset{\sim}{I}(\underset{\sim}{k}, \underset{\sim}{\ell}) \,|\, \underset{\sim}{b}) = T \tag{3}$$

In fact, it can be verified that the data given in Table 1 of MR-60 obey the corresponding rules for SNSLOG. Thus the condition $\underline{\underline{a}} \supseteq s(k)$ found there, has the property given in (4) below and the right hand side is analogous to the conditions employed in the above rules (1), (2), (3).

$$(\underline{\underline{a}} \supseteq s(k)) \subseteq (\underline{\underline{a}} \cap s(k) \neq \emptyset) \tag{4}$$

The conditions $\underset{\sim}{a} \cap q(\underset{\sim}{k}) \neq \emptyset$ and $\underset{\sim}{a} \cap q(\underset{\sim}{k}^{c}) \neq \emptyset$ are expressible in MATLOG as

$$\text{QSCPDT(QVA,QK)} = 1 \ , \quad \text{QSCPDT(QVA,QCOMP(QK))} = 1 \tag{5}$$

We shall not prove the above rules (1), (2), (3) for 3x3 matrices in QLOGIC, but prove them for the even more general case of m x n matrices of GLOGIC, employing clausal relations, which are the analogues of the equations employed in QLOGIC. This will be done by showing that the Boolean truth value

$$\mathrm{QBTV(QVA,QMZ,QVB)} \equiv \mathrm{QBINPT(QVA,QMZ,CVB)} \tag{6}$$

for QMZ = QMO, QK, QL, differs from

$$\begin{aligned}&\mathrm{QBTV2(QVA,QMO,QK,QL,QVB)}\\ &\quad\equiv \mathrm{BSUM(QSCPDT(QVA,QK),QSCPDT(QVB,QL))}\end{aligned} \tag{7}$$

for precisely the above conditions, namely

$$\begin{aligned}&\mathrm{QVA} = \mathrm{Q8}, \quad \mathrm{QSCPDT(QVB,QL)} = 1\\ \text{or}\quad &\\ &\mathrm{QSCPDT(QVA,QK)} = 1, \quad \mathrm{QVB} = \mathrm{Q8}\end{aligned} \tag{8}$$

The analogues in QL-2 of Problems 2 and 3 of SNS in Part I will not be given here, but will be briefly taken up at the end of this report, after discussing the implementation of elementary relations in PLOGIC(QL-1A) and PQLOG (QL-1B).

These are to be contrasted with a quantified relation expressed in QL-1. In a logical relation of the type QL-1, the relation really exists between the _individual_ components $\underset{\sim}{a}x$ and $\underline{\underline{b}}x$ for _each_ value of the variable x, and both $\underline{\underline{a}}x$ and $\underline{\underline{b}}x$ have the same domain of operation. This is particularly made clear by the form of the relation in QL-1 in general, namely

$$(\underset{\sim}{q_Z}x)(\underline{\underline{a}}x \ \underline{\underline{Z}}\ \underline{\underline{b}}x) \qquad (2)$$

where there is only a single variable x contained in the relation and the form of its unary and binary implementation is as in (3a), (3b) below:

Unary : $\underset{\sim}{a}x\ ,\ (\underset{\sim}{q_Z}x)(\underline{\underline{a}}x\ \underline{\underline{Z}}\ \underline{\underline{b}}x) \vdash\!\!\Rightarrow \underset{\sim}{b}x$ (3a)

Binary : $\underset{\sim}{a}x,\ \underset{\sim}{b}x,\ (\underset{\sim}{q_Z}x)(\underline{\underline{a}}x\ \underline{\underline{Z}}\ \underline{\underline{b}}x) \vdash\!\!\Rightarrow \underline{\underline{c}}$, SNS truth value of the relation for the given inputs. (3b)

A few simple examples of unary and binary relations are given in (4a-d) and (5a-c) below. The unary examples are

$$(\forall x)(\underline{\underline{a}}x),\ (\exists x)(\underline{\underline{a}}x \Longrightarrow \underline{\underline{b}}x) \longmapsto (\exists x)(\underline{\underline{b}}x)$$

$$(\exists x)(\underline{\underline{a}}x),\ (\exists x)(\underline{\underline{a}}x \Longrightarrow \underline{\underline{b}}x) \longmapsto (\Delta x)(\underline{\underline{b}}x) \qquad (4a\text{-}d)$$

$$(\exists x)(\underline{\underline{a}}x),\ (\forall x)(\underline{\underline{a}}x \wedge \underline{\underline{b}}x) \longmapsto (\forall x)(\underline{\underline{b}}x)$$

$$(\exists x)(\underline{\underline{a}}x),\ (\exists x)(\underline{\underline{a}}x \wedge \underline{\underline{b}}x) \longmapsto (\exists x)(\underline{\underline{b}}x)$$

Similarly, we have the binary examples

$$(\forall x)(\underline{\underline{a}}x),\quad (\forall x)(\neg \underline{\underline{b}}x),\quad (\exists x)(\underline{\underline{a}}x \wedge \underline{\underline{b}}x) \longmapsto \underline{\underline{c}} = F$$

$$(\exists x)(\underline{\underline{a}}x),\quad (\exists x)(\neg \underline{\underline{b}}x),\quad (\forall x)(\underline{\underline{a}}x \wedge \neg \underline{\underline{b}}x) \longmapsto \underline{\underline{c}} = D \qquad (5a\text{-}c)$$

$$(\exists x)(\underline{\underline{a}}x),\quad (\forall x)(\underline{\underline{b}}x),\quad (\exists x)(\underline{\underline{a}}x \Longrightarrow \underline{\underline{b}}x) \longmapsto \underline{\underline{c}} = T$$

As will be seen from the form of the expressions in (3a) and (3b), both vectors representing quantifiers, as well as SNS terms, occur in these formulae, and it is not apparent as to how they can be converted into BVMF containing essentially 3-vectors. However, as has been briefly outlined in MR-56, Lecture-4, this class of equations, which has been named QL-1A, can be formulated in a compact and unified treatment, by first considering the quantifier algebra QL-1, which is effectively equivalent to QL-1A for the particular case when $(\underset{\sim}{q}_Z x) = (\forall x)$. Then Eqs. 3(a,b) respectively go over into 6(b) and 6(a) below. We shall discuss these and the nature of their relationship with 3(a,b) below.

$$\underset{\sim}{a}x,\ \underset{\sim}{b}x,\ \underset{\sim}{a}x \ \underline{\underline{Z}}\ \underset{\sim}{b}x = \underset{\sim}{c}x \longmapsto \underset{\sim}{c}x \quad \text{(Binary forward)} \qquad (6a)$$

$$\underset{\sim}{c}x,\ \underset{\sim}{a}x,\ \underset{\sim}{a}x \ \underline{\underline{Z}}\ \underset{\sim}{b}x = \underset{\sim}{c}x \longmapsto \underset{\sim}{b}x \quad \text{(Binary reverse)} \qquad (6b)$$

As has been described in MR-52, 53 and 54, the QL-1 type of relation occurs from the existence of a logical relation between the individual components $\underline{\underline{a}}x$ and $\underline{\underline{b}}x$ of the corresponding members of the sets A and B having the quantifier states $\underset{\sim}{a}x$ and $\underset{\sim}{b}x$, which lead to $\underline{\underline{c}}x$ of the same individual x,

and hence to the resultant set C described by the quantifier state $\underset{\sim}{cx}$. Thus the implementation of the binary relation (7)

$$(\exists x)(\underline{\underline{a}}x),\ (\forall x)(\underline{\underline{b}}x),\ \underset{\sim}{ax} \wedge \underset{\sim}{bx} = \underset{\sim}{cx} \qquad (7)$$

leads to $\underset{\sim}{cx} = (\exists x)(\underline{\underline{c}}x)$ for the following reasons. We are given that "at least one member " $(\exists x)$ of the full set $\mathcal{S}$ has the property of the set A, and that "all members" $(\forall x)$ of the set $\mathcal{S}$ have the property of the set B, and we wish to find out the quantified nature of the members of $\mathcal{S}$ that have the property of both Set A and Set B. Denoting this set by C and the corresponding vector by $\underset{\sim}{cx}$, it is obvious that we can only say that "there exists at least one member" $(\exists x)$ of $\mathcal{S}$ that have the property of the set C. Thus a binary relation in QL-1 has the individual output $\underline{\underline{c}}x$ in an SNS state, but this leads to a quantifier state represented by a 3-vector $\underset{\sim}{cx}$ which is of the same type as the quantifier states of the two inputs $\underset{\sim}{ax}$ and $\underset{\sim}{bx}$.

The principle of the procedure discussed above can be used for working out the resultant quantifier state $\underset{\sim}{cx}$ of the set C which is the logical sum, or product, of the quantifier states $\underset{\sim}{ax}$ and $\underset{\sim}{bx}$, and more generally for a relation $\underline{\underline{Z}}(k, \mathcal{L})$ connecting

$\underline{\underline{a}}x$ and $\underline{\underline{b}}x$ of sets A and B for their quantified states $\underset{\sim}{a}$ and $\underset{\sim}{b}$ being any one of the eight standard states q(1) to q(8). This has been done ab initio and 3x3 tables for these operators have been obtained.

Table 4. 3x3 tables for the QL-1 connectives "and" and "or"

(a) AND ($\underset{=}{A}$)

$\underset{=}{A}$	$\forall$	Σ	Φ
$\forall$	$\forall$	Σ	Φ
Σ	Σ	Λ	Φ
Φ	Φ	Φ	Φ

(b) OR ($\underset{=}{O}$)

$\underset{=}{O}$	$\forall$	Σ	Φ
$\forall$	$\forall$	$\forall$	$\forall$
Σ	$\forall$	$\exists$	Σ
Φ	$\forall$	Σ	Φ

The detailed mathematical properties of these tables are discussed in MR-53,54, and a short summary given in MR-56, Lecture 4. Here, we shall only indicate those formulae which are relevant for working out the computer-implementable algorithms given below. As will be seen from Table 4(a, b), the outputs for the QL-1 relations $\underset{\sim}{ax}\ \underset{=}{A}\ \underset{\sim}{bx}$ and $\underset{\sim}{ax}\ \underset{=}{O}\ \underset{\sim}{bx}$ for the generator states q(1), q(3), q(5) have very simple representation in lattice algebra. Without going into this lattice theory as such, we may merely write the effective

algebraic formulae for $\underset{\sim}{c}x$ in terms of $\underset{\sim}{a}x$ and $\underset{\sim}{b}x$ as follows.

Denoting the states of $\underset{\sim}{a}$, $\underset{\sim}{b}$ and $\underset{\sim}{c}$ as $q(i)$, $q(j)$ and $q(k)$, we have

<u>QL-1 -- "and"</u>

$$\underset{\sim}{q}(i)\ \underline{\underline{A}}\ \underset{\sim}{q}(j) = \underset{\sim}{q}(k)$$

where

$k =$ Max (i, j), and $k = 2$ if $i = j = 3$

and $k = 8$ if i or $j = 8$

<u>QL-1 -- "or"</u>

$$\underset{\sim}{q}(i)\ \underline{\underline{O}}\ \underset{\sim}{q}(j) = \underset{\sim}{q}(k)$$

where

$k =$ Min (i, j), and $k = 6$ if $i = j = 3$

and $k = 8$ if i or $j = 8$

The extension of these 3x3 tables into 8x8 tables is obtained from noting that, a general quantifier state, $q(n)$, $n = 1$ to 8, is the Boolean sum of at most three basic states $q(1)$, $q(3)$, $q(5)$. We therefore make use of the analogs of the formula (8) given below, for obtaining the QL-1 conjunction or disjunction of these mixed states. Thus, for $\underset{\sim}{a} = q(i_1) \oplus q(i_2)$ and $\underset{\sim}{b} = q(j_1) \oplus q(j_2)$, $\underset{\sim}{a}\ \underline{\underline{A}}\ \underset{\sim}{b} = \underset{\sim}{c}$ is

$$\underset{\sim}{c} = (q(i_1) \oplus q(i_2))\ \ \underline{\underline{A}}\ \ (q(j_1) \oplus q(j_2))$$

$$= (q(i_1)\ \underline{\underline{A}}\ q(j_1)) \oplus (q(i_2)\ \underline{\underline{A}}\ q(j_1)) \oplus (q(i_1)\ \underline{\underline{A}}\ q(j_2))$$

$$(q(i_2)\ \underline{\underline{A}}\ q(j_2)) \quad (8$$

The tables so obtained for $\underline{\underline{Z}} = \underline{\underline{A}}(1, 1)$ and $\underline{\underline{O}}(1, 1)$ are given in page 12 of Lecture-4, MR-56. They will be reproduced as computer outputs later in this report.

The natural question arises as to how we should proceed to formulate the algorithms for the connectives $\underline{\underline{I}}$ and $\underline{\underline{E}}$ and also for a general logical connective $\underline{\underline{Z}}(k, \ell)$. Both these are possible by noting that in QL-1, the following general formula holds for $k, \ell = 1, 2$.

$$\underset{\sim}{ax}\ \underline{\underline{Z}}(k, \ell)\ \underset{\sim}{bx} \;=\; \underset{\sim}{a}'x\ \underline{\underline{Z}}(1, 1)\ \underset{\sim}{b}'x \qquad (9)$$

where

$$\underset{\sim}{a}'x = \underset{\sim}{ax},\ \underset{\sim}{a}^{n}x \quad \text{according as } k = 1,\ 2$$

$$\underset{\sim}{b}'x = \underset{\sim}{bx},\ \underset{\sim}{b}^{n}x \quad \text{according as } \ell = 1,\ 2$$

The main point to be noted is that the appropriate operators that are to be applied to $\underset{\sim}{ax}$ and $\underset{\sim}{bx}$, depending on the value of $k, \ell = 1$ or 2, are the matrix-Boolean operators QEQU and QNOT respectively. Particularly, the occurrence of the operator $\underset{\sim}{N}$ is to be noted for QL-1, as contrasted with the operator $\underset{\sim}{M}$ which occurs in similar situations in QL-2. The theoretical reason for this is the fact that it is the predicate $\underline{\underline{ax}}$ in $(qx)(\underline{\underline{a}}x) = \underset{\sim}{ax}$ that is negated and this produces from $\neg\underline{\underline{ax}}$ the quantifier state $\underset{\sim}{ax}\ \underset{\sim}{N}$ which is the same as the MATLOG function QNOT(QVA).

With this introduction to the algebraic theory, we shall formulate some of the essential functions required in PLOGIC which are to be formulated in the QL-1 algebra described above.

(b) MATLOG functions for QL-1

The two operators QEQU and QNOT are particularly needed in PLOGIC as they occur very frequently in it. Also, two functions, namely QELEM and QVECEL, have to be explicitly stated although they have been used in all the subroutines in QLOGIC. These are given in (iii).

(iii) Decomposition of a vector QVA into its basic vectors QELEM

QELEM(QVA, QBAS,IB)

This program takes in as input a vector QVA having components VA(1), VA(3), VA(5) which are given by VECEL(QVA), and gives as output one, two, or three, basic vectors, out of (1 0 0), (0 1 0), (0 0 1) — which are components of it — as QBAS(I), I = 1, IB, where IB is the number of basic vectors contained in QVA. It has the following structure.

```
                QVECEL(QVA) = VA(3)
                       IB = 0

   If   VA(1) = 1,   IB = IB + 1,    QBAS(IB) = Q1
   If   VA(2) = 1,   IB = IB + 1,    QBAS(IB) = Q3
   If   VA(3) = 1,   IB = IB + 1,    QBAS(IB) = Q5
```

Further, to take care of the lattice properties, two subroutines QMIN and QMAX are also needed.

(iv) Calculation of the minimum of two basic vectors QMIN

This function has the form QMIN(QVA,QVB), where QVA = QI and QVB = QJ, are the inputs. The output is QMIN = QK, where K = min (I,J), the function min standing for the arithmetical minimum of I and J.

(v) Determination of the maximum of two vectors QMAX

As with QMIN, this has the form QMAX(QVA,QVB). With the same inputs QI, QJ, QMAX = QK, where K = max(I,J).

It is to be noted that the minimum and maximum have to do only with the indexes of the Q-vectors QVA,QVB in the standard form Q1 to Q8 adopted by us, and the only values that occur for I, J, K are 1, 3 and 5, since these functions are required only for basic vectors.

Unlike QLOGIC, there are not two different operators $\underset{\sim}{U}$, $\underset{\sim}{O}$ and $\underset{\sim}{V}$, $\underset{\sim}{A}$ in PLOGIC, but a combination of both types of functions occurs in the QL-1 operators "or" and "and" as mentioned above. However, these two are given the MATLOG names PUNION and PVIDYA and the algebraic properties mentioned above can be converted into the subroutines given below.

(vi) <u>QL-1 "or"</u> — ax $\underset{=}{O}$ bx = cx : PUNION

```
          PUNION(QVA,QVB)  =  QVC
          QELEM(QVA)  = QBAS1(IB1)
          QELEM(QVB)  = QBAS2(IB2)
      IF  (QVA = Q8) . OR . (QVB = Q8)  QOUT = Q8
          GO TO  1
      ELSE   QOUT = Q8
      DO     I = 1, IB1,  J = 1, IB2
10        QOUT = QUNION(QOUT,QMIN(QBAS1(I), QBAS2(J))
          GO TO  1
1         QVC = QOUT
```

(vii) <u>QL-1 "and"</u> — ax $\underset{=}{A}$ bx = cx : PVIDYA

```
          PVIDYA(QVA,QVB) = QVC
          Same as PUNION except that QMIN is replaced
          by QMAX  in  10  as
20        QOUT = QUNION(QOUT,QMAX(QBAS1(I), QBAS2(J))
```

Employing these two QL-1 connectives, the other two connectives "imply" and "equivalent" can be simply defined in QL-1 as in (10a), (10b). Hence no separate functions are defined for these and they are calculable from the general subroutine PBINPT given in (viii) below.

$$q(i) \underset{=}{I} q(j) = q(i^{n}) \underset{=}{O} q(j) \quad ; \tag{10a}$$

$$q(i) \underset{=}{E} q(j) = (q(i) \underset{=}{A} q(j)) \underset{=}{O} (q(i^{n}) \underset{=}{A} q(j^{n})) \tag{10b}$$

(The corresponding Eq.(8) on page 11 of Lecture-4 is to be replaced by (10a) and (10b) as being more precise.)

Combining (9), for dealing with a general $\underline{\underline{Z}}(k, \ell)$, with (10) to extend the theory from $\underline{\underline{A}}(1, 1)$ and $\underline{\underline{O}}(1, 1)$ to $\underline{\underline{E}}(1, 1)$ and $\underline{\underline{I}}(1, 1)$, the algorithm for a general relation $\underset{\sim}{a}x\ \underline{\underline{Z}}(k, \ell)\ \underset{\sim}{b}x$ in the binary form of implementation can be made, as given in the algorithm for the function PBlNPT below. As mentioned earlier, the output of this binary product in QL-1 is a 3-vector and not a Boolean scalar as in SNS and QL-1. Consequently, the function has the form given in (viii) below.

(viii) <u>QL-1 Binary product</u> — $\underset{\sim}{a}x\ \underline{\underline{Z}}(k, \ell)\ \underset{\sim}{b}x = \underset{\sim}{c}x$: PBINPT

```
        PBINPT(QVA,SMX,SK,SL,QVB) = QVC

         SK = 1  --> QOK = PEQU
         SK = 2  --> QOK = PNOT
         SL = 1  --> QOL = PEQU
         SL = 2  --> QOL = PNOT

    QVC = PVIDYA(QOK(QVA), QOL(QVB))     for SMX = SMA
    QVC = PUNION(QOK(QVA), QOL(QVB))     for SMX ≙ SMO
    QVC = PUNION(PNOT(QOK(QVA)), QOL(QVB)) for SMX = SMI
    QVC = PUNION(PVIDYA(QOK(QVA),QOL(QVB)) ,
          PVIDYA(PNOT(QOK(QVA)), PNOT(QOL(QVB))))  for  SMX = SME
```

The above are the subroutines which are required under QL-1, which is not practically applied as such, but theoretically very valuable for QL-1A calculus which is the standard form

employed in predicate calculus. It will be noticed that a general SNS matrix SMZ is not defined in our QL-1 algebra, but only the logical matrices $\underline{\underline{Z}}(k,\ell)$ in SNS, namely SMX,SK,SL with X = $\underline{\underline{A}}$, $\underline{\underline{O}}$, $\underline{\underline{E}}$, $\underline{\underline{I}}$, are employed in the subroutines mentioned above for PLOGIC. In fact, only these matrices are used in the definition of logical relations in standard predicate calculus which is our QL-1A algebra. (The extension to all 16 2x2 matrices $\underline{\underline{Z}}$ can be done, but has not been included here, since a very different formalism for implementing PLOGIC may have to be employed for this purpose.) However, it should be mentioned that the formulae given above are the analogs of SSTV and QSTV, rather than SSTV2 and QSTV2, since as will be seen from the outputs in Section 4(e) , the output of a binary relation with $\underset{\sim}{a} = \emptyset$, or $\underset{\sim}{b} = \emptyset$ in $\underset{\sim}{a}\, \underline{\underline{Z}}\, \underset{\sim}{b} = \underset{\sim}{c}$ is always $\underset{\sim}{c} = \emptyset$ for both $\underline{\underline{A}}$-type and $\underline{\underline{O}}$-type connectives $\underline{\underline{Z}}$, (and naturally also for $\underline{\underline{E}}$-type and $\underline{\underline{I}}$-type). As will be seen from the practical examples to be worked out later, there is complete consistency between logical relations, and we really deal with logical relations in quantifier theory, and not with merely 3-member set theory. This will become clearer after examining particular examples of the application of the theory.

Before going to the algorithms in QL-1A, we shall give a brief introduction to the algebraic theory of the relation between QL-1 and QL-1A formulations and then give the algorithms for obtaining the output of unary relations and the SNS truth values of binary relations in QL-1A.

(c) Theory of the QL-1A binary relation

The equations that we will deal with are (3a) and (3b) for unary and binary relations in QL-1A, which correspond respectively to (6b) and (6a) of QL-1 for binary reverse and binary forward relations, as mentioned in Section 4(a). We shall now give the essential features of this relationship which enables us to carry over the algorithms developed above for QL-1 almost completely into QL-1A.

Considering first, the QL-1A binary relation (3b), we are given a relation, whose connective is associated with a quantifier $(\underset{\sim}{q}_Z x)$ and an SNS connective $\underline{\underline{Z}}(k,\ell)$, and we wish to find out the truth value of this relation for the inputs $\underset{\sim}{a}x$ and $\underset{\sim}{b}x$. In QL-1, we have already worked out the subroutine PBINPT to work out the quantifier state $\underset{\sim}{c}x$ of the resultant term when $\underset{\sim}{a}x$ and $\underset{\sim}{b}x$ are related by $\underline{\underline{Z}}(k,\ell)$ as in (6a). To obtain (3b) from this, it is only necessary to find out the relative truth value QRELTV of $\underset{\sim}{c}x$ for $\underset{\sim}{q}_Z x$. We shall illustrate this by the second example

of QL-1A binary relation in (5b). In this, the inputs are $\underset{\sim}{ax} = (\exists x)(\underline{\underline{a}}x) = (1\ 1\ 0)$, $\underset{\sim}{bx} = (\exists x)(\neg\underline{\underline{a}}x) = (0\ 1\ 1)$, which are put in the relation $\underset{\sim}{ax}\ \underline{\underline{A}}(1,\ 2)\ \underset{\sim}{bx}$, and can be shown, using FBINPT, to yield the output $\underset{\sim}{cx} = (1\ 1\ 1) = (\bigwedge x)(\underline{\underline{c}}x)$. We wish to obtain the truth value of this for the quantifier state $(\forall x)$ in Eq.(5b). For this, we make use of QL-2 algebra, and calculate $\underline{\underline{t}}(\exists | \forall) = (\langle 1\ 1\ 1 | 1\ 0\ 0\rangle,\ \langle 1\ 1\ 1 | 0\ 1\ 1\rangle)$ $= (1\ 1) = D$, which is the answer given in (5b). Looking at the problem logically, if $(\exists x)(\underline{\underline{a}}x)$ is true, and $(\exists x)(\neg\underline{\underline{b}}x)$ is true, then we cannot say for certain that $\underline{\underline{c}}x = \underline{\underline{a}}x \wedge \neg\underline{\underline{b}}x$ is true for at least one x, since x_1 for which $\underline{\underline{a}}x$ is true, may, or may not, correspond with x_2 for which $\underline{\underline{b}}x$ is true. On the other hand, the input conditions also permit that both $\underline{\underline{a}}x$ and $\neg\underline{\underline{b}}x$ could be true for all x_1 so that the quantifier state $(\forall x)(\underline{\underline{c}}x)$ is also possible. Hence the relation $(\forall x)(\underline{\underline{a}}x \wedge \neg\underline{\underline{b}}x)$ could be either true or false, i.e. it is indefinite for the given inputs, so that its truth value is the SNS state (1 1) = D.

If, on the other hand, in (5b) the second input is $(\exists x)(\underline{\underline{b}}x) = (1\ 1\ 0)$, then using the same subroutine FBINPT, we obtain $\underset{\sim}{cx} = (0\ 1\ 1) = (\exists x)(\neg\underline{\underline{c}}x)$. The relative truth value (QRELTV) of this for $(\forall x)(\underline{\underline{c}}x)$ is $\underline{\underline{t}}(\bigwedge | \forall) = (\langle 0\ 1\ 1 | 1\ 0\ 0\rangle,$ $\langle 0\ 1\ 1 | 0\ 1\ 1\rangle) = (0\ 1\ ) = F$, which indicates that the inputs definitely do not satisfy the relation. Logically, also, if for some x, $\underline{\underline{b}}x$ is true, then the expression $(\forall x)(\underline{\underline{a}}x \wedge \neg\underline{\underline{b}}x)$ which require $\underline{\underline{b}}x$ to be false for all x, cannot be true, so that the truth value of the relation for the given inputs $(\exists x)(\underline{\underline{a}}x)$ and $(\exists x)(\underline{\underline{b}}x)$ is $F = (0\ 1)$. It is interesting that all these considerations are automatically taken care of in EVMF.

Thus, the algorithm for calculating the SNS truth value of a QL-1A relation becomes straightforward and may be given as follows in subroutine (ix) PSTV.

(ix) QL-1A binary truth value -- $(\underset{\sim}{q}_Z x)(\underset{\sim}{a}x\ \underline{\underline{Z}}(k, \ell)\ \underset{\sim}{b}x) = \underline{\underline{c}}x$: PSTV

```
PSTV(QVZ,QVA,SMX,SK,SL,QVB) =  SVC
QVC = PBINPT(QVA,SMX,SK,SL,QVB)
SVC = QRELTV(QVC,QVZ)
```

This is probably the only algorithm that is needed for binary relations in QL-1A algebra. We shall now consider unary relations of the type (3a).

(d) Theory of the QL-1A unary relation

Before considering the QL-1A unary relation (3a), we shall consider Eq.(6b) giving the QL-1 binary reverse relation. The principle of the binary reverse has been discussed in several earlier reports, and as applied to QL-1, it is considered in some detail in MR-53 and 54. Essentially, the idea is that the relation $\underset{\sim}{a}x\ \underline{\underline{Z}}\ \underset{\sim}{b}x = \underset{\sim}{c}x$ is given, and instead of calculating $\underset{\sim}{c}x$ as output for given $\underset{\sim}{a}x$ and $\underset{\sim}{b}x$, we calculate $\underset{\sim}{b}x$ as output for given $\underset{\sim}{c}x$ and $\underset{\sim}{a}x$. The word "reverse" is applied because

the output of the relation is available, and in reverse, it restricts the relationship between $\underset{\sim}{a}x$ and $\underset{\sim}{b}x$, so that, if one is given, the other can be calculated. If this idea is kept in mind, then 3x3 Tables 4(a) and 4(b) can be reversed to obtain $\underset{\sim}{b}$, given $\underset{\sim}{c}$ and $\underset{\sim}{a}$, as in Table 5(a) and 5(b) (copied from MR-56). Note that the notation for the reverse relation $\underset{\sim}{a}\ (\underset{\sim}{c}, \underset{\sim}{Z}) = \underset{\sim}{b}$ corresponds to the forward relation $\underset{\sim}{a}\ \underset{=}{Z}\ \underset{\sim}{b} = \underset{\sim}{c}$.

Table 5. 3x3 truth table for QL-1 binary reverse relations

(a) $\underset{\sim}{a}(\underset{\sim}{c}, \underset{=}{A}) = \underset{\sim}{b}$

$\underset{\sim}{a}$ \ $\underset{\sim}{c}$	∀	Σ	Φ
∀	∀	Σ	Φ
Σ	∅	∃	Λ
Φ	∅	∅	Δ

(b) $\underset{\sim}{a}(\underset{\sim}{c}, \underset{=}{O}) = \underset{\sim}{b}$

$\underset{\sim}{a}$ \ $\underset{\sim}{c}$	∀	Σ	Φ
∀	Δ	∅	∅
Σ	∃	Λ	∅
Φ	∀	Σ	Φ

These tables have been derived in MR-53 and 54. We shall indicate here how they are algorithmised for application. As given in page 15, Lecture-4 of MR-56, a very neat algorithm can be given for the quantifier state of the output as given in Eqs. (11a,b) and (12a,b) below. Denoting the quantifier states of $\underset{\sim}{a}$ and $\underset{\sim}{c}$ by q(i) and q(k), then, for i,k = 1, 3,5 for basic states, the quantifier state q(j), of the output $\underset{\sim}{b}$, is given by (11)and (12) respectively, for the connectives $\underset{=}{A}(1, 1)$ and $\underset{=}{O}(1, 1)$.

For $\underline{\underline{A}}$

$$\text{If} \quad k < i \,, \quad q(j) = q(8) \tag{11a}$$

$$\text{If} \quad k \geqslant i, \; q(j) = \bigoplus_{\ell = 0}^{L} q(k - i + 1 + 2\;), \text{ where } \; L = (i+1)/2 \tag{11b}$$

For $\underline{\underline{O}}$

$$\text{If} \quad k > i \,, \quad q(j) = q(8) \tag{12a}$$

$$\text{If} \quad k \leqslant i, \; q(j) = \bigoplus_{\ell = 0}^{L} q(k + 2\;), \text{ where } \; L = (7 - i)/2 \tag{12b}$$

To obtain $\underset{\sim}{a}(\underset{\sim}{c},\, \underline{\underline{I}}) = \underset{\sim}{b}$, we make use of the equivalence $(\underset{\sim}{a}\; \underline{\underline{I}}\; \underset{\sim}{b} = \underline{\underline{c}}) \Longleftrightarrow (\underset{\sim}{a}^n\; \underline{\underline{O}}\; \underset{\sim}{b} = \underline{\underline{c}})$, so that, in reverse, we can express the operation of $\underline{\underline{I}}$ by $\underline{\underline{O}}$, as in (13).

$$(\underset{\sim}{a}(\underset{\sim}{c},\, \underline{\underline{I}}) = \underset{\sim}{b}) \Longleftrightarrow (\underset{\sim}{a}^n(\underset{\sim}{c},\, \underline{\underline{O}}) = \underset{\sim}{b}) \tag{13}$$

Also, more generally, for $\underline{\underline{A}}(k, \ell)$ and $\underline{\underline{O}}(k, \ell)$, we use (9) for the forward relation, which yields in reverse, the equations (14a,b,c).

$$(\underset{\sim}{a}(\underset{\sim}{c},\, \underline{\underline{Z}}(k, \ell)) = \underset{\sim}{b}) \Longleftrightarrow (\underset{\sim}{a}'(\underset{\sim}{c},\, \underline{\underline{Z}}(1,1)) = \underset{\sim}{b}') \tag{14a}$$

where

$$\underset{\sim}{a}' = \underset{\sim}{a},\; \underset{\sim}{a}^n \quad \text{according as} \quad k = 1,\; 2 \;\; ; \tag{14b}$$

$$\underset{\sim}{b}' = \underset{\sim}{b},\; \underset{\sim}{b}^n \quad \text{according as} \quad \ell = 1,\; 2 \tag{14c}$$

No simple expression can be given for $\underline{\underline{E}}$, occurring in a QL-1 reverse relation, in terms of the corresponding relations for $\underline{\underline{A}}$ and $\underline{\underline{O}}$, in this formalism. However, by directly inverting

the 3x3 truth table of the forward relation $\underset{\sim}{a}\ \underline{\underline{E}}\ \underset{\sim}{b} = \underset{\sim}{c}$, it can be shown that the reverse relation $\underset{\sim}{a}(\underset{\sim}{c}, \underline{\underline{E}}) = b$ has exactly the same 3x3 table in the format of Tables 5(a) and 5(b), as the corresponding table for the forward relation in the format of Tables 4(a) and 4(b). (See MR-53 and MR-56 for these tables -- the corresponding 8x8 tables are printed out in Section 4(e)).

We shall now show that the binary reverse relation in QL-1 is equivalent to the unary relation in QL-1A, by taking an example, as was done for the binary forward relation. The inter-relationship between (3a) and (6b) is that, if $\underset{\sim}{c}x$ of (6b) is put equal to $(\underset{\sim}{q_Z}x)$ of (3a), then the two equations are logically equivalent to one another. Considering the example (4a), we are given that there exists at least one x'' for which $\underline{\underline{a}}x$ implies $\underline{\underline{b}}x$, and that $\underline{\underline{a}}x$ is true for all x''. Then the conclusion that "there exists an x'' for which $\underline{\underline{b}}x$ is true is obvious. In the QL-1 formalism, it takes the following form. In the binary forward form, given that ax is true for all x, and bx is true for at least one x , the relation $\underset{\sim}{a}x \Longrightarrow \underset{\sim}{b}x = \underset{\sim}{c}x$ is true for at least one x. Therefore, given $\underline{\underline{a}}x$ is true for all x,

and that the relation is true for at least one x, what is the quantifier state of $\underline{\underline{b}}x$ that will satisfy this condition? Obviously, it is $(\exists x)(\underline{\underline{b}}x)$.

Hence, Eqs.(11a,b) and (12a,b) can be taken over bodily for Eq.(3a) with the equivalences (15), for all $\underline{\underline{Z}} = \underline{\underline{A}}, \underline{\underline{O}}, \underline{\underline{E}}, \underline{\underline{I}}$:

$$(\underset{\sim}{a} = q(i),\ \underset{\sim}{b} = q(j),\ \underset{\sim}{c} = q(k)) \text{ of QL-1}$$
$$\Longleftrightarrow (\underset{\sim}{a} = q(i),\ \underset{\sim}{b} = q(j),\ \underset{\sim}{q_Z} = q(k)) \text{ of QL-1A} \qquad (15)$$

The two subroutines for $\underline{\underline{A}}$ and $\underline{\underline{O}}$ are named "unary and", "unary or" of PLOGIC, and refer to QL-1A algebra with the inputs put in the sequence $\underset{\sim}{q_Z}$, $\underset{\sim}{a}$, and the output being labelled $\underset{\sim}{b}$. The following MATLOG subroutines will become obvious from the above considerations.

(x) <u>Unary and for QL-1A</u> — $\underset{\sim}{a}x,\ (\underset{\sim}{q_Z}x)(\underline{\underline{a}}x\ \underline{\underline{A}}\ \underline{\underline{b}}x) \longmapsto \underset{\sim}{b}x$: PUNAND

```
            PUNAND(QVZ,QVA) = QVB
            QELEM(QVA) = QBAS1(IB1)
            QELEM(QVZ) = QBAS2(IB2)
   IF   QVZ . OR . QVA = Q8,  QOUT = Q8,   GO TO  1
   ELSE         QOUT = Q8
            DO  I1 = 1, IB2,  J1 =  1, IB1
```

```
        Denote   QBAS2(I1)   as   QK
        Denote   QBAS1(J1)   as   QI
        If  $K < I$,   QJ = Q8
        else   $QJ = Q(K - I + 1) \oplus Q(K - I + 3) \oplus Q(K - I + 5)$
        where number of terms is $(I + 1)/2$
10      QOUT = QUNION(QOUT,QJ)
 1      QVB  = QOUT
```

(xi) Unary or for QL-1A — $\underset{\sim}{a}x$, $(\underset{\sim}{q}_Zx)(\underline{\underline{a}}x\ \underline{\underline{O}}\ \underline{\underline{b}}x) \mapsto bx$: PUNOR

PUNOR(QVZ,QVA) = QVB

Same as for PUNAND except that the conditions are replaced by

```
        If  $K > I$,   QJ = Q8
        else   $QJ = Q(K - I + 1) \oplus Q(K - I + 3) \oplus Q(K - I + 5)$
        where number of terms is $(7 - I)/2$
```

We can generalize these two unary relations to a general unary relation of the type in (3a) with $\underline{\underline{Z}} = \underline{\underline{Z}}(k, \ell)$, $Z = A, O, E$ or I. In an obvious notation, this takes the form of the function PUNPDT below.

(xii) General unary product for QL-1A — $\underset{\sim}{a}x$, $(\underset{\sim}{q}_Zx)(\underline{\underline{a}}x\ \underline{\underline{Z}}(k, \ell)\ \underline{\underline{b}}x) \mapsto \underset{\sim}{b}x$: PUNPDT

PUNPDT(QVZ,QVA,SMX,SK,SL) = QVB

```
SK = S1  -->  QOK  =  QEQU
SK = S2  -->  QOK  =  QNOT
SL = S1  -->  QOL  =  QEQU
SL = S2  -->  QOL  =  QNOT
```

```
If  SMX = SMA ,
        QVBP = PUNAND(QVZ,QOK(QVA))
        QVB  = QOL(QVBP)

If  SMX = SMO ,
        QVB =  QOL(PUNOR(QVZ,QOK(QVA)))

If  SMX = SMI  ,
        QVB = QOL(PUNOR(QVZ, QOK(PNOT(QVA)))

If  SMX = SME  ,
        QVB = PUNION(PVIDYA(QOK(QVA),QOL(QVZ)),
              PVIDYA(PNOT(QOK(QVA)),PNOT(QOL(QVZ))))
```

This completes all the subroutines that are needed for PLOGIC in its application to QL-1 type binary forward and binary reverse relations as well as QL-1A type unary and binary relations. No separate program is written for QL-1 binary reverse relations as they are equivalent to QL-1A unary relations and these subprograms for PUNAND and PUNOR can be used for this purpose with QVZ being replaced by QVC for the QL-1 binary relation.

As will be seen from the outputs of these subroutines which are obtained as solutions of some of the problems given below, the MATLOG algorithms, which are based on a semi-empirical algebraic notation employed for PLOGIC, agree completely with the truth tables obtained from an intuitive logical approach in the earlier reports MR- 53, 54 and 56.

(e) Illustrations and problems in PLOGIC

The examples given below are of three types.

(a) Outputs of the subroutines PUNION, PVIDYA, PUNAND PUNOR, PBINPT , PUNPDT and PSTV. Not all of them are printed out, but some examples are shown to illustrate the nature of the output of these functions.

(b) Some formal relations between different logical connectives are tested out in 8x8 tables by the application of MATLOG. These include the description of equivalence as the union of suitable conjunctions and as the vidya of suitable implications, the De Morgan relation between a conjunction and its corresponding disjunction in QL-1, and the equivalence (13) in QL-1A :

$$(\underset{\sim}{q_Z^{\ell}}x)(\underline{\underline{a}}x\ \underline{\underline{Z}}^{c}\ \underline{\underline{b}}x) \Longleftrightarrow \neg(\underset{\sim}{q_Z}x)(\underline{\underline{a}}x\ \underline{\underline{Z}}\ \underline{\underline{b}}x) \qquad (13)$$

(c) Illustrative examples of some simple problems in predicate calculus solved by PLOGIC subroutine of MATLOG.

Problems 9A and B

A. Obtain the 8x8 truth table for binary "or" and "and" in QL-1, employing the MATLOG subroutines PUNION and PVIDYA.

B. Similarly obtain the 8x8 truth table for the connectives "imply" and "equivalent" using PBINPT.

Comments on the output of Problems 9A and 9B

The four tables corresponding to 9A and 9B are given below in Tables 5(a-d) on pages 45 and 46. It can be verified that they agree completely with Tables 7(a-d), pages 120, 121 of MR-53 which were obtained by intuitively working out the logic of each QL-1 binary relation from ab initio considerations based on the definition of the QL-1 relation in terms of the input truth values of ax, bx and cx.

Problems 10 A and 10B

Both the functions PUNOR and PUNAND as well as the general subroutine PUNPDT can be checked by obtaining a set of tables similar to Problem 9 from the MATLOG programs. This is done via problems 10A and B.

10A : Print out the 8x8 truth tables for QL-1 "or"(PUNOR) and QL-1 "and"(PUNAND) for.the logical relation $(\underset{\sim}{q}_Z x)(ax\ Z = bx)$ for $Z = A, O$.

Table 5. Outputs of the subroutines PUNOR, PUNAND and of PUNPDT for E(1, 1) and I(1, 1)

(a)
```
OUTPUT OF PUNOR

      OPR   ALL   EXS   NAL   NEX   SOM   IND   AON   IMP
OPA

ALL         ALL   ALL   ALL   ALL   ALL   ALL   ALL   IMP

EXS         ALL   EXS   EXS   EXS   EXS   EXS   EXS   IMP

NAL         ALL   EXS   IND   NAL   EXS   IND   IND   IMP

NEX         ALL   EXS   NAL   NEX   SOM   IND   AON   IMP

SOM         ALL   EXS   EXS   SOM   EXS   EXS   EXS   IMP

IND         ALL   EXS   IND   IND   EXS   IND   IND   IMP

AON         ALL   EXS   IND   AON   EXS   IND   AON   IMP

IMP         IMP   IMP   IMP   IMP   IMP   IMP   IMP   IMP
```

(b)
```
OUTPUT OF PUNAND

      OPR   ALL   EXS   NAL   NEX   SOM   IND   AON   IMP
OPA

ALL         ALL   EXS   NAL   NEX   SOM   IND   AON   IMP

EXS         EXS   IND   NAL   NEX   NAL   IND   IND   IMP

NAL         NAL   NAL   NAL   NEX   NAL   NAL   NAL   IMP

NEX         NEX   NEX   NEX   NEX   NEX   NEX   NEX   IMP

SOM         SOM   NAL   NAL   NEX   NAL   NAL   NAL   IMP

IND         IND   IND   NAL   NEX   NAL   IND   IND   IMP

AON         AON   IND   NAL   NEX   NAL   IND   AON   IMP

IMP         IMP   IMP   IMP   IMP   IMP   IMP   IMP   IMP
```

(c)

```
      OUTPUT FOR PRINT OF I(1,I)

           QVB   ALL   EXS   NAL   NEX   SOM   IND   AON   IMP
 QVA

 ALL             ALL   EXS   NAL   NEX   SOM   IND   AON   IMP

 EXS             EXS   IND   IND   NAL   IND         IND   IMP

 NAL             NAL   IND   IND   EXS   IND   IND   IND   IMP

 NEX             NEX   NAL   EXS   ALL   SOM   IND   AON   IMP

 SOM             SOM   IND   IND   SOM   IND   IND   IND   IMP

 IND             IND   IND   IND   IND   IND   IND   IND   IMP

 AON             AON   IND   IND   AON   IND   IND   AON   IMP

 IMP             IMP   IMP   IMP   IMP   IMP   IMP   IMP   IMP
```

(d)

```
      OUTPUT FOR PRINT OF  I(1,I)

           QVB   ALL   EXS   NAL   NEX   SOM   IND   AON   IMP
 QVA

 ALL             ALL   EXS   NAL   NEX   SOM   IND   AON   IMP

 EXS             ALL   EXS   IND   NAL   EXS   IND   IND   IMP

 NAL             ALL   EXS   EXS   EXS   EXS   EXS   EXS   IMP

 NEX             ALL   ALL   ALL   ALL   ALL   ALL   ALL   IMP

 SOM             ALL   EXS   EXS   SOM   EXS   EXS   EXS   IMP

 IND             ALL   EXS   IND   IND   EXS   IND   IND   IMP

 AON             ALL   EXS   IND   AON   EXS   IND   AON   IMP

 IMP             IMP   IMP   IMP   IMP   IMP   IMP   IMP   IMP
```

10B : Print the truth tables for the above predicate calculus relation for the connectives E(1, 1), I(1, 1), A(2, 2), O(2, 2) using the general subroutine PUNPDT for these connectives.

Comments on the output of Problems 10A and 10B

The outputs are shown in Tables 6(a-d) in pages 48 and 49. It will be noticed that the tables are not symmetric in the two inputs, namely QVZ and QVA, since the former is the quantifier q_Z associated with the connective Z , while the latter is the input ax. It can be checked that the first two tables 6(a,b) for the Problem 10A are identical with Tables 6(a,b) in page 113 of MR-53, obtained for the same problems by intuitive *ab initio* calculations, using the logic of the relation between ax and bx for individual members of the set that are quantified. The data in Tables 6(c,d) of Problem 10B can be seen to agree with Tables 6(c,d) in page 113a of MR-53, while those in (e, f) indicate how other connectives can be implemented.

The fact that simple algorithms could be formulated for working out all the possibilities indicate the cogency and consistency of the algebraic formalism which has been adopted for predicate logic *via* the BA-3 algebra of quantifiers. The tests given below are designed to check the interconnections between these formulae.

Table 6. 8x8 truth tables for unary relations in QL-1 with the connectives A(1, 1), O(1, 1), E(1, 1) and I(1, 1)

(a) OUTPUT OF FUNOR

QVA QVZ	ALL	EXS	NAL	NEX	SOM	IND	AON	IMP
ALL	IND	IND	EXS	ALL	EXS	IND	IND	IMP
EXS	IND	IND	IND	EXS	IND	IND	IND	IMP
NAL	IMP	NAL	NAL	NAL	NAL	NAL	NAL	IMP
NEX	IMP	IMP	NEX	NEX	IMP	NEX	NEX	IMP
SOM	IMP	NAL	NAL	SOM	NAL	NAL	SOM	IMP
IND	IND	IND	IND	IND	IND	IND	IND	IMP
AON	IND	IND	IND	AON	EXS	IND	IND	IMP
IMP	IMP	IMP	IMP	IMP	IMP	IMP	IMP	IMP

(b) OUTPUT OF FUNAND

QVA QVZ	ALL	EXS	NAL	NEX	SOM	IND	AON	IMP
ALL	ALL	ALL	IMP	IMP	IMP	ALL	ALL	IMP
EXS	EXS	EXS	EXS	IMP	EXS	EXS	EXS	IMP
NAL	NAL	IND	IND	IND	IND	IND	IND	IMP
NEX	NEX	NAL	IND	IND	NAL	IND	IND	IMP
SOM	SOM	EXS	EXS	IMP	EXS	EXS	SOM	IMP
IND	IND	IND	IND	IND	IND	IND	IND	IMP
AON	AON	IND	IND	IND	NAL	IND	IND	IMP
IMP	IMP	IMP	IMP	IMP	IMP	IMP	IMP	IMP

```
(c) OUTPUT FOR PUNPDT OF E(1,1)

     QVA  ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
QVZ

ALL       ALL  EXS  NAL  NEX  SOM  IND  AON  IMP

EXS       EXS  IND  IND  NAL  IND  IND  IND  IMP

NAL       NAL  IND  IND  EXS  IND  IND  IND  IMP

NEX       NEX  NAL  EXS  ALL  SOM  IND  AON  IMP

SOM       SOM  IND  IND  SOM  IND  IND  IND  IMP

IND       IND  IND  IND  IND  IND  IND  IND  IMP

AON       AON  IND  IND  AON  IND  IND  AON  IMP

IMP       IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

```
(d) OUTPUT FOR PUNPDT OF I(1,1)

     QVA  ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
QVZ

ALL       ALL  EXS  IND  IND  EXS  IND  IND  IMP

EXS       EXS  IND  IND  IND  IND  IND  IND  IMP

NAL       NAL  NAL  NAL  IMP  NAL  NAL  NAL  IMP

NEX       NEX  NEX  IMP  IMP  IMP  NEX  NEX  IMP

SOM       SOM  NAL  NAL  IMP  NAL  NAL  SOM  IMP

IND       IND  IND  IND  IND  IND  IND  IND  IMP

AON       AON  IND  IND  IND  EXS  IND  IND  IMP

IMP       IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

(e)
```
OUTPUT FOR FUNPDT OF A(2,2)

      QVA  ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
 QVZ

 ALL       ALL  ALL  IMP  IMP  IMP  ALL  ALL  IMP

 EXS       EXS  EXS  EXS  IMP  EXS  EXS  EXS  IMP

 NAL       NAL  IND  IND  IND  IND  IND  IND  IMP

 NEX       NEX  NAL  IND  IND  NAL  IND  IND  IMP

 SOM       SOM  EXS  EXS  IMP  EXS  EXS  SOM  IMP

 IND       IND  IND  IND  IND  IND  IND  IND  IMP

 AON       AON  IND  IND  IND  NAL  IND  IND  IMP

 IMP       IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

(f)
```
OUTPUT FOR FUNPDT OF O(2,2)

      QVA  ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
 QVZ

 ALL       IND  IND  EXS  ALL  EXS  IND  IND  IMP

 EXS       IND  IND  IND  EXS  IND  IND  IND  IMP

 NAL       IMP  NAL  NAL  NAL  NAL  NAL  NAL  IMP

 NEX       IMP  IMP  NEX  NEX  IMP  NEX  NEX  IMP

 SOM       IMP  NAL  NAL  SOM  NAL  NAL  SOM  IMP

 IND       IND  IND  IND  IND  IND  IND  IND  IMP

 AON       IND  IND  IND  AON  EXS  IND  IND  IMP

 IMP       IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

Problem 11A, B

A : Verify that the two ways of writing the equivalence relation, as disjunction of $\underline{\underline{A}}(1,1)$ and $\underline{\underline{A}}(2,2)$, and the conjunction of $\underline{\underline{I}}(1,1)$ and $\underline{\underline{I}}(2,2)$, which is valid in SNS, can be carried over to QL-1. This can done by writing the equivalence relation $\underset{\sim}{ax}\ \underline{\underline{E}}(1,1)\ \underset{\sim}{bx}$ as

$$(\underset{\sim}{ax}\ \ \underline{\underline{A}}(1,\ 1)\ \underset{\sim}{bx})\ \ \underline{\underline{O}}(1,\ 1)\ \ (\underset{\sim}{ax}\ \ \underline{\underline{A}}(2,\ 2)\ \ \underset{\sim}{bx}) \qquad (16a)$$

and

$$(\underset{\sim}{ax}\ \ \underline{\underline{I}}(1,\ 1)\ \underset{\sim}{bx})\ \ \underline{\underline{A}}(1,\ 1)\ \ (\underset{\sim}{ax}\ \ \underline{\underline{I}}(2,\ 2)\ \ \underset{\sim}{bx}) \qquad (16b)$$

B : In a similar manner, check the following De Morgan relation given in (17), analogous to the De Morgan relation in SNS, between the connectives $\underline{\underline{O}}(1,\ 2)$ and $\underline{\underline{A}}(1,\ 2)$:

$$(\underset{\sim}{ax}\ \ \underline{\underline{O}}(1,\ 2)\ \ \underset{\sim}{bx})\ \ =\ \ \neg(\neg\,\underset{\sim}{ax}\ \ \underline{\underline{A}}(1,\ 2)\neg\,\underset{\sim}{bx}) \qquad (17)$$

Solution of the Problems 11A and B

A: The MATLOG statements to be verified are as follows:

```
        QVC1 = PUNION(PVIDYA(QVA,QVB),
               PVIDYA(QNOT(QVA),QNOT(QVB)))

        QVC2 = PVIDYA(PIMPLY(QVA,QVB), PIMPLY(QNOT(QVA),QNOT(QVB)))
where          PIMPLY(QVA,QVB) = PUNION(QNOT(QVA),QVB )
```

The 8x8 tables for QVC1 and QVC2 obtained as computer outputs are given below in Table 7, page 51.

Table 7. Output for Problem 11A

```
                      QVC1

     QVB    ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
 QVA

 ALL        ALL  EXS  NAL  NEX  SOM  IND  AON  IMP

 EXS        EXS  IND  IND  NAL  IND  IND  IND  IMP

 NAL        NAL  IND  IND  EXS  IND  IND  IND  IMP

 NEX        NEX  NAL  EXS  ALL  SOM  IND  AON  IMP

 SOM        SOM  IND  IND  SOM  IND  IND  IND  IMP

 IND        IND  IND  IND  IND  IND  IND  IND  IMP

 AON        AON  IND  IND  AON  IND  IND  AON  IMP

 IMP        IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

```
                           QVC2

     QVB    ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
 QVA

 ALL        ALL  EXS  NAL  NEX  SOM  IND  AON  IMP

 EXS        EXS  IND  IND  NAL  IND  IND  IND  IMP

 NAL        NAL  IND  IND  EXS  IND  IND  IND  IMP

 NEX        NEX  NAL  EXS  ALL  SOM  IND  AON  IMP

 SOM        SOM  IND  IND  SOM  IND  IND  IND  IMP

 IND        IND  IND  IND  IND  IND  IND  IND  IMP

 AON        AON  IND  IND  AON  IND  IND  AON  IMP

 IMP        IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

It will be seen that the two tables completely agree and also that they agree with Table 7(d) of MR-53, where the 8x8 table, obtained from apriori logical considerations, is given.

11B : The MATLOG statements for verifying the equivalence of the r.h.s and l.h.s of (17) are

```
PBINPT(QVA,SMO,S1,S2,QVB)  =  QVC1

QNOT(PBINPT(QNOT(QVA),SMA,S1,S2, QNOT(QVB))) = QVC2
```

The check that QVC1 = QVC2 for all possible quantifier states, Q1 to Q8, for the inputs QVA and QVB, may be seen from Table 8 below. This indicates the essential facility of MATLOG algorithms to take care any general logical statement in QL-1. It should be noted that "negation" is QNOT, corresponding to the BA-3 operator $\underset{\sim}{N}$, for QL-1A logic, and this is uniformly applicable to all formulae. The reason for this is because the negation is for the predicate and not for the quantifier as in QL-2 logic.

Table 8. Output for Problem 11B

```
                         QVC1

     QVB   ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
  QVA

  ALL      ALL  ALL  ALL  ALL  ALL  ALL  ALL  IMP

  EXS      ALL  EXS  EXS  EXS  EXS  EXS  EXS  IMP

  NAL      ALL  EXS  IND  NAL  EXS  IND  IND  IMP

  NEX      ALL  EXS  NAL  NEX  SOM  IND  AON  IMP

  SOM      ALL  EXS  EXS  SOM  EXS  EXS  EXS  IMP

  IND      ALL  EXS  IND  IND  EXS  IND  IND  IMP

  AON      ALL  EXS  IND  AON  EXS  IND  AON  IMP

  IMP      IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

```
                        QVC2

   QVB   ALL  EXS  NAL  NEX  SOM  IND  AON  IMP
QVA

ALL      ALL  ALL  ALL  ALL  ALL  ALL  ALL  IMP

EXS      ALL  EXS  EXS  EXS  EXS  EXS  EXS  IMP

NAL      ALL  EXS  IND  NAL  EXS  IND  IND  IMP

NEX      ALL  EXS  NAL  NEX  SOM  IND  AON  IMP

SOM      ALL  EXS  EXS  SOM  EXS  EXS  EXS  IMP

IND      ALL  EXS  IND  IND  EXS  IND  IND  IMP

AON      ALL  EXS  IND  AON  EXS  IND  AON  IMP

IMP      IMP  IMP  IMP  IMP  IMP  IMP  IMP  IMP
```

Problem 12A, B

These two problems require a new function PSMXCP, which is described in Section 4(f) in pages 56 and 57 which should be read before going further. These two problems will be based on Eq.(13) on page 43 and will check the validity of the equivalence given therein for chosen inputs. Problem 12A will check PBINPT and PSTV, while 12B will check PUNPDT.

12A : Check the equality of the resultant truth values $\underline{\underline{c}}_1$ and $\underline{\underline{c}}_2$ of the two equivalent statements (18a,b) for the 8x8 possible pairs $\underset{\sim}{a}x$, $\underset{\sim}{b}x$.

$$\neg(\exists x)(\underline{\underline{a}}x\ \underline{\underline{I}}\ \underline{\underline{b}}x) \iff (\forall x)(\underline{\underline{a}}x\ \underline{\underline{I}}^c\ \underline{\underline{b}}x) \qquad (18a,b)$$

Solution to Problem 12A

The MATLOC statements for this can be given as follows:

```
PSTV(QCOMP(Q6),QVA,SMI,S1,S1,QVB)  =  SVC1
PSTV(Q1,QVA,PSMXCP(SMI,S1,S1),QVB) = SVC2
```

Input QVA,QVB = Q1 to Q8 .

The equality of SVC1 and SVC2 may be seen from Tables 9(a, b) giving the outputs for SVC1 and SVC2 .

Table 9 : 8x8 Tables of SVC1 and SVC2 for PSTV

SVC1

QVB QVA	ALL	EXS	NAL	NEX	SOM	IND	AON	IMP
ALL	F	F	D	T	F	D	D	X
EXS	F	F	D	D	F	D	D	X
NAL	F	F	F	F	F	F	F	X
NEX	F	F	F	F	F	F	F	X
SOM	F	F	F	F	F	F	F	X
IND	F	F	D	D	F	D	D	X
AON	F	F	D	D	F	D	D	X
IMP	X	X	X	X	X	X	X	X

QVB QVA	ALL	EXS	NAL	NEX	SOM	IND	AON	IMP
ALL	F	F	D	T	F	D	D	X
EXS	F	F	D	D	F	D	D	X
NAL	F	F	F	F	F	F	F	X
NEX	F	F	F	F	F	F	F	X
SOM	F	F	F	F	F	F	F	X
IND	F	F	D	D	F	D	D	X
AON	F	F	D	D	F	D	D	X
IMP	X	X	X	X	X	X	X	X

Problem 12B

Check the equality of the quantifier states of the two outputs $\underset{\sim}{b_1}x$ and $\underset{\sim}{b_2}x$ given by (19a,b) for all the eight possible quantifier states of $(\underset{\sim}{q_z}x)$ and $\underset{\sim}{a}x$.

$$\underset{\sim}{a}x,\quad \neg(\underset{\sim}{q_z}x)(\underline{\underline{a}}x\ \underline{\underline{I}}\ \underline{\underline{b}}x)\ \mapsto \underset{\sim}{b_1}x \tag{19a}$$

$$\underset{\sim}{a}x,\quad (\underset{\sim}{q_z^{\ell}}x)(\underline{\underline{a}}x\ \underline{\underline{I}}^{c}\ \underline{b}x)\ \mapsto \underset{\sim}{b_2}x \tag{19b}$$

Solution to Problem 12B

The Fortran statements for the outputs QVB1 and QVB2, in terms of the inputs QVZ and QVA, are as follows:

```
FUNPDT(QCOMP(QVZ),QVA,SMI,S1,S1) = QVB1
FUNPDT(QELL(QVZ),QVA,PSMXCP(SMI,S1,S1)) = QVB2
```

For checking the equality of QVB1 and QVB2, the 8x8 tables are printed side by side in Table 10(a, b). It will be seen that the two tables agree, showing that the algorithms including the new function PSMXCP are well formulated.

Table 10 : 8x8 tables for QVB1 and QVB2 of Problem 12B

QVB1

QVA QVZ	ALL	EXS	NAL	NEX	SOM	IND	AON	IMP
ALL	NAL	NAL	NAL	IMP	NAL	NAL	NAL	IMP
EXS	NEX	NEX	IMP	IMP	IMP	NEX	NEX	IMP
NAL	ALL	EXS	IND	IND	EXS	IND	IND	IMP
NEX	EXS	IND	IND	IND	IND	IND	IND	IMP
SOM	AON	IND	IND	IND	EXS	IND	IND	IMP
IND	IMP	IMP	IMP	IMP	IMP	IMP	IMP	IMP
AON	SOM	NAL	NAL	IMP	NAL	NAL	SOM	IMP
IMP	IND	IND	IND	IND	IND	IND	IND	IMP

QVB2

QVA QVZ	ALL	EXS	NAL	NEX	SOM	IND	AON	IMP
ALL	NAL	NAL	NAL	IMP	NAL	NAL	NAL	IMP
EXS	NEX	NEX	IMP	IMP	IMP	NEX	NEX	IMP
NAL	ALL	EXS	IND	IND	EXS	IND	IND	IMP
NEX	EXS	IND	IND	IND	IND	IND	IND	IMP
SOM	AON	IND	IND	IND	EXS	IND	IND	IMP
IND	IMP	IMP	IMP	IMP	IMP	IMP	IMP	IMP
AON	SOM	NAL	NAL	IMP	NAL	NAL	SOM	IMP
IMP	IND	IND	IND	IND	IND	IND	IND	IMP

(f) Complementation of the name of an SNS logical connective

The standard definition of the SNS truth value of a relation $\underline{\underline{a}}\ \underline{\underline{Z}}(k, \ell)\ \underline{\underline{b}} = \underline{\underline{c}}$ is based on the two equations $\langle \underline{\underline{a}} \,|\, \underline{\underline{Z}}(k, \ell) \,|\, \underline{\underline{b}} \rangle = c_1$, $\langle \underline{\underline{a}} \,|\, \underline{\underline{Z}}^c(k, \ell) \,|\, \underline{\underline{b}} \rangle = c_2$, leading to $\underline{\underline{c}} = (c_1, c_2)$. For doing this, the procedure adopted so far is to convert the 2x2 matrix $\underline{\underline{Z}}(k, \ell)$ into its complement by taking the Boolean complement of each of the components $Z_{\lambda\mu}$ of the matrix. In other words, the definition of the complement of a matrix connective is the one associated with the Boolean complement of its matrix. This, and the associated formulae for 3x3 matrices in QLOGIC, have been the main basis of all the formulae so far. However, in PLOGIC and also in PQLOGIC, we find the need to associate the <u>name</u> $\underline{\underline{Z}}(k, \ell)$ with the <u>designation</u> of $\underline{\underline{Z}}$ as one of $\underline{\underline{A}}, \underline{\underline{O}}, \underline{\underline{E}}, \underline{\underline{I}}$ and the <u>indices</u> k or ℓ standing for S1 or S2, and these are to be transformed into the <u>name</u> and <u>indices</u> of the complement of the connective. Thus, $\underline{\underline{A}}^c(1, 1) = \underline{\underline{O}}(2, 2)$; $\underline{\underline{I}}^c(1, 1) = \underline{\underline{O}}^c(2, 1) = \underline{\underline{A}}(1, 2)$. Denoting the original connective by $\underline{\underline{P}}$, and its complement $\underline{\underline{P}}^c$ by $\underline{\underline{Q}}$, we wish to associate the name P, and the indices k_P and ℓ_P , with the name Q and the indices k_Q and ℓ_Q by a suitable functiom.

This is readily possible, in terms of our MATLOG notation, by defining a new function called "name-complement" of an SNS matrix in PLOGIC, namely PSMXCP, having the following structure.

(xiii) Name-complement of an SNS connective in PLOGIC -- $\underline{\underline{P}}^{c} = \underline{\underline{Q}}$

Denoting $\underline{\underline{P}}$ by SMXP,SKP,SLP and $\underline{\underline{Q}}$ by SMXQ,SKQ,SLQ, we have

PSMXCP(SMXP,SKP,SLP) = SMXQ,SKQ,SLQ

For SMXP = SMA, SMXQ = SMO, SKQ = SCOMP(SKP), SLQ = SCOMP(SLP)

SMXP = SMO, SMXQ = SMA, SKQ = SCOMP(SKP), SLQ = SCOMP(SLP)

SMXP = SMI, SMXQ = SMA, SKQ = SKP, SLQ = SCOMP(SLP)

SMXP = SME, SMXQ = SME, SKQ = SKP, SLQ = SCOMP(SLP)

As mentioned in connection with Problems 12A and 12B, this subroutine has interesting possibilities for solving problems in PLOGIC and PQLOG, using the algebra of QL-1A and QL-1B. It can be side stepped and incorporated in the program for various functions as given below. However, the function PSMXCP is very valuable for theoretical analysis of the inter-relations between QL-2, QL-1 QL-1A and QL-1B, and the program is likely to find application in checking such formulae. In fact, the discussion given in the next Section 4(g) arose as a result of testing such ideas in connection with the algorithmization of QL-1B.

(g) Construction of the matrix of a "logical" connective in QL-2

As mentioned above, when the above ideas were fed in for PQLOGIC (see Section 5), we observed that there is a need for a

function for obtaining the matrix $|\underset{\sim}{Z}(\underset{\sim}{k}, \underset{\sim}{\ell})|$ in GL-2 for a relation of the type $(\forall x)(\underline{a}x)\ \underline{O}(2, 1)\ (\exists y)(\underline{b}y)$, from the three given data, namely $(\forall x), \underline{O}(2, 1), (\exists y)$, which has not been formulated as an algorithm in MATLOG, although it is implied in (17) of Section 1(c) dealing with logical operators in GL-2. In the above example, the SNS connective $\underline{O}(2, 1)$, standing for $\neg\underline{a} \vee \underline{b}$, is converted, by addition of the quantifiers $(\forall x)$ and $(\exists y)$ to $\underline{a}$ and $\underline{b}$, into the form $\neg(\forall x)(\underline{a}x) \vee (\exists y)(\underline{b}y)$ corresponding to the GL-2 connective $\underset{\sim}{O}(\underset{\sim}{2}, \underset{\sim}{6})$. The feature that requires attention, from the point of view of the algorithm, is that the operation of negation" in SNS (SCOMP) in $\neg\underline{a} \vee \underline{b}$ is converted into the operation of "complementation" in GL-2 (QBM) in $\neg(\forall x)(\underline{a}x)$, and the latter is attached to the quantifier $\forall$ which is associated with $\underline{a}$, and not to the predicate $\underline{a}$. This may look obvious, but it has to be properly introduced into the set of subroutines, for it finds application in various problems. In fact, in an analogous situation in GL-1B, $(\forall x)(\exists y)(\neg\underline{a}x \vee \underline{b}y)$ has the equivalent form $(\forall x)(\neg\underline{a}x) \vee (\exists y)(\underline{b}y)$ in GL-2, which corresponds to the connective $\underset{\sim}{O}(\underset{\sim}{5}, \underset{\sim}{6})$ and the SNS "negation" SCOMP in $\neg\underline{a}x$ is converted into the

"negation" of the quantified term in QL-2, corresponding to the operation QMBN. The full algebraic theory of these is derivable from the material presented in MR-53 and MR-54, but it is found that they take a much simpler algorithmic form which has been discovered in the attempt at computerizing them. A brief summary of the theory of QL-1B, and the algorithm for the implementation of elementary statements in QL-1B *via* their QL-2 equivalents, are presented in Section 5 below. The treatment here closely follows the notation adopted in MR-56, Lecture-4, for QL-1A and which we have adopted for PLOGIC. Since the form of the relation in QL-1B is similar to QL-1, while it is implemented by converting it into QL-2, these types of statements are **put** under the title "PQLOGIC" and discussed in the next Sections **5(a,b).** The procedure for obtaining the equivalent matrix connective is closely similar to that in PLOGIC and the corresponding algorithm is given in **Section 5(b).**

In view of these, we give below the amended form of Q(xvii.) as Q(xviia) and Q(xvii b), the first to obtain the name of the matrix QMZ,QK,QL in terms of QVK,SMX,SK,SL,QVL,

and the second for obtaining the matrix elements of the 3x3 matrix corresponding to QMZ,QK,QL. We give, in Section 5(c), the subroutine PQ(i), corresponding to Q(xviia), as applied to PQLOGIC.

(xviia) Matrix connective in QLOGIC defined via an SMS logical connective: $\underset{\sim}{q}(\underset{\sim}{k}_Z)\ \underset{=}{Z}(k,\ell)\ \underset{\sim}{q}(\underset{\sim}{\ell}_Z) = \underset{\sim}{Z}(\underset{\sim}{k},\underset{\sim}{\ell})$

QMCON(QVK,SMX,SK,SL,QVL) = QMZ,QK,QL

The notation may be illustrated by the example given above, for which we have

QMCON(Q1, SMI,S2,S1,Q6) = QMI,Q2,Q6

The statements required for this subroutines are

QMZ = QMA,QMO,QMI,QME according as SMX = SMA,SMO,SMI,SME

QK = QEQU(QVK),QNOT(QVK) according as SK = S1,S2

QL = QEQU(QVL),QNOT(QVL) according as SL = S1,S2

(xviib) Matrix elements of a QL-2 logical connective:

$|\underset{\sim}{Z}(\underset{\sim}{k},\underset{\sim}{\ell})| = Z_{ij}$, i = 1,2,3, j = 1,2,3

QMATEL(QMZ,QK,QL) = QM(I, J)

If QMZ = QMA, QM = QDIRPT(QK,QL)

If QMZ = QMO, QM = QDIRSM(QK,QL)

If QMZ = QMI, QM = QDIRSM(QCOMP(QK),QL)

If QMZ = QME, QM = QDIREQ(QK,QL)

5. Multiple quantifiers and their implementation

(a) Elementary statement in QL-1B

Just as QL-1A employs a quantifier associated with the logical connective, for a statement composed of relations between quantifier states, in the form $(\underset{\sim}{q}_Z x)(\underline{\underline{a}}x\ \underline{\underline{Z}}(k, \ell)\ \underline{\underline{b}}x)$, the QL-1B type has the standard form,

$$(\underset{\sim}{q}_Z x)(\underset{\sim}{q}_Z y)\ (\underline{\underline{a}}x\ \underline{\underline{Z}}(k, \ell)\ \underline{\underline{b}}y) \qquad (1)$$

Typical examples are

$$(\forall x)(\exists y)\ (\underline{\underline{a}}x \Longrightarrow \underline{\underline{b}}y) \qquad (2a)$$

$$(\exists x)(\exists y)\ (\underline{\underline{a}}x \wedge \underline{\underline{b}}y) \qquad (2b)$$

In our notation and formulation of predicate logic, statements of the type (2a) and (2b) have a very specific interpretation which is somewhat different from the standard one ; but all standard relations can be shown to be expressible, in one form or the other, in terms of QL-2, QL-1A and QL-1B forms.

Thus, in (2a) and (2b), the variables x and y associated with $\underset{\sim}{a}$ and $\underset{\sim}{b}$ are <u>different</u> in QL-1B, and the quantifier

associated with each is given by the variable that is adopted for the corresponding quantifier. Otherwise, there is no significance attached to the sequence of the quantifiers. Thus for example

$$2a) \equiv (\exists y)(\forall x)(\underline{\underline{a}}x \Rightarrow \underline{\underline{b}}y) \qquad (3)$$

It becomes particularly obvious when this is written in terms of the disjunction operator as

$$(2a) \equiv (\forall x)(\exists y)(\neg \underline{\underline{a}}x \vee \underline{\underline{b}}y) \equiv (\forall x)(\exists y)(\underline{\underline{b}}y \vee \neg \underline{\underline{a}}x) \qquad (4a)$$

$$\equiv \quad (3) \quad \equiv (\exists y)(\forall x)(\neg \underline{\underline{a}}x \vee \underline{\underline{b}}y) \qquad (4b)$$

In Eqs. (4 a) and (4 b), there is no first term and second term within the bracket since they are connected by the commutative operator "or", and no difference can be associated with the sequence of the quantifiers, which are different in (4 a) and (4 b). This feature will become particularly clear when we give below the algorithms based on the essential principle of the standard prenex normal form of quantifier calculus. This principle is, however, applied in the reverse sense in BVMF, and the QL-1B form is converted into an equivalent QL-2 statement, which is then implemented by the techniques of Section 2. We shall indicate this below, and only the essential formulae, required for writing the computer programs, are given. A fuller treatment will be given in a separate report.

(b) Implementation of QL-1B statements in predicate logic via PQLOG.

The name PQLOG is applied to this sub-program because it implements P-type statements (QL-1B), but <u>via</u> the process of converting them into QLOGIC statements (QL-2). The subroutines for QLOGIC and PLOGIC, given in Sections 3 and 4 above, are assumed to be available, although they may not all be required. It is found that, just as in PLOGIC, we consider only a notation and formalism for describing a PQLOG relation and its unary and binary forms of implementation, for the <u>standard</u> SKS <u>logical connectives</u> of the type $\underline{\underline{A}}$, $\underline{\underline{O}}$, $\underline{\underline{I}}$, $\underline{\underline{E}}$, and not for a general 2x2 matrix. The relevant algebraic formulae, which are computerized in Section (c), are given below.

(i) QL-1B binary relation

A binary relation, in general, has the form

$$\underset{\sim}{a}x,\ \underset{\sim}{b}y,\ (\underset{\sim}{q}_{\underset{\sim}{Z}}x)(\underset{\sim}{q}_{\underset{\sim}{Z}}y)(\underline{\underline{a}}x\ \underline{\underline{Z}}(k, \ell)\ \underline{\underline{b}}y) \longmapsto \underline{\underline{c}} \qquad (5a)$$

which is equivalent to the QL-2 form

$$\underset{\sim}{a}x,\ \underset{\sim}{b}y,\ \ \underset{\sim}{a}x\ \underset{\sim}{Z}'(\underset{\sim}{k}, \underset{\sim}{\ell})\ \underset{\sim}{b}y \longmapsto \underline{\underline{c}} \qquad (5b)$$

where

$$Z' = Z\ ,\ \text{for}\ \ A,\ O \qquad (5c)$$

and in (5b),

$$q(\underset{\sim}{k}) = \underset{\sim}{q}_{\underset{\sim}{Z}}x \text{ or } \underset{\sim}{q}^{n}_{\underset{\sim}{Z}}x\ ,\ \text{according as } k = 1,\ 2 \text{ in } (5a) \qquad (6a)$$

$$q(\underset{\sim}{\ell}) = \underset{\sim}{q}_{\underset{\sim}{Z}}y \text{ or } \underset{\sim}{q}^{n}_{\underset{\sim}{Z}}y\ ,\ \text{according as } \ell = 1,2 \text{ in } (5a) \qquad (6b)$$

Extensions of these for $Z = I, E$ are available, by using the relation between implication and disjunction, and the description of equivalence as the disjunction of $\underline{\underline{A}}(1, 1)$ and $\underline{\underline{A}}(2, 2)$, as in QL-1A.

Thus, $I(1, 1) = O(2, 1)$, so that (5c) continues to be valid, $Z' = Z = I$, but (6a) gets changed to

$$q(\underset{\sim}{k}) = \underset{\sim}{q}{}_Z^{\ell}x \text{ or } \underset{\sim}{q}{}_Z^{m}x \text{ , according as } k = 1, 2 \text{ in (5a)} \qquad (6c)$$

leaving (6b) unaltered.

Similarly, for $\underline{\underline{Z}}(k, \ell) = \underline{\underline{E}}(k, \ell)$, $k, \ell = 1, 2$, the relation (5a) becomes

$$(\underset{\sim}{q}{}_Zx)(\underset{\sim}{q}{}_Zy)\ ((\underline{\underline{a}}x\ \underline{\underline{A}}(k, \ell)\ \underline{\underline{b}}y) \bigvee (\underline{\underline{a}}x\ \ \underline{\underline{A}}(k^c, \ell^c)\ \ \underline{\underline{b}}y)) \qquad (7a)$$

$$= (\underset{\sim}{a}x\ \underset{\sim}{A}(\underset{\sim}{k}, \underset{\sim}{\ell})\ \underset{\sim}{b}y) \bigvee (\underset{\sim}{a}x\ \ \underset{\sim}{A}(\underset{\sim}{k}'\ , \underset{\sim}{\ell}')\ \ \underset{\sim}{b}y) \qquad (7b)$$

where $q(\underset{\sim}{k}), q(\underset{\sim}{k}') = \underset{\sim}{q}{}_Zx$ or $\underset{\sim}{q}{}_Z^nx$, according as $k, k^c = 1, 2$ (7c)

$q(\underset{\sim}{\ell}), q(\underset{\sim}{\ell}') = q_Zy$ or q_Z^ny, according as $\ell, \ell^c = 1, 2$ (7d)

Eqs.(7b,c,d) are valid <u>only</u> for the interpretation (7a) of (5a) for $\underline{\underline{E}}(k, \ell)$, and <u>not</u> for $\underline{\underline{E}}$ expressed as the conjunction of two implications.

(ii) QL-1B unary relation

This has the form

$$\underset{\sim}{a}x,\ (q_Z x)(q_Z y)(\underline{\underline{a}}x\ \underline{\underline{Z}}(k,\ell)\ \underline{\underline{b}}y) \mapsto \text{by} \qquad (8a)$$

which takes the equivalent QL-2 form

$$\underset{\sim}{a}x,\ \underset{\sim}{a}x\ \underset{\sim}{Z}'(\underset{\sim}{k},\underset{\sim}{\ell}) \mapsto \underset{\sim}{b}y \qquad (8b)$$

and the relation between Z and Z' follows what was given above for the binary relation, with the change that, for $\underline{\underline{E}}(k,\ell)$

$$\underset{\sim}{a}x,\ \underset{\sim}{a}x\ \underset{\sim}{E}(\underset{\sim}{k},\underset{\sim}{\ell}) = (\underset{\sim}{a}x\ \underset{\sim}{A}(\underset{\sim}{k},\underset{\sim}{\ell}) \bigcirc\!\!\!\!\!\oplus\ \underset{\sim}{a}x\ \underset{\sim}{A}(\underset{\sim}{k}',\underset{\sim}{\ell}')) \mapsto \underset{\sim}{b}y \qquad (8c)$$

Essentially, the only new FORTRAN formulae that are needed for PQLOG are those for the unary, and binary products, and the SNS truth value of a binary relation, in PQLOG. These three functions PQBNPT, PQSTV, PQUNPT are described below as PQ(ii)--(iv). (These are preceded by PQMCON, in the algorithm PQ(i), analogous to QMCON in Q(xviia)).

(c) Subroutines in PQLOG

PQ(i) : Matrix connective in QLOGIC corresponding to a logical relation in PQLOG defined via an SNS logical connective : $\underset{\sim}{q}(\underset{\sim}{k}_Z)\ \underset{\sim}{q}(\underset{\sim}{\ell}_Z)\ (\underline{\underline{a}}\ \underline{\underline{Z}}(k,\ell)\ \underline{\underline{b}}) = \underset{\sim}{Z}(\underset{\sim}{k},\underset{\sim}{\ell})$

PQMCON(QVK,SMX,SK,SL,QVL) = QMZ,QK,QL

For SMX = SMA,SMO, QMZ = QMA,QMO

QK = QEQU(QVK),QNOT(QVK) according as SK = S1,S2

QL = QEQU(QVL),QNOT(QVL) according as SL = S1,S2

For SMX = SMI , QMZ = QMI

QK = QEQU(QVK), QCOMP(QVK) according as SK = S1,S2

QL = QEQU(QVL), QNOT(QVL) according as SL = S1,S2

No equivalent single logical QL-2 connective is available for the case of SMX = SME. However, the corresponding unary and binary relations can be implemented via Eqs 7(a-d) and (8c) given above in Section 5(b).

Thus, for SMX = SME

PQMCON(QVK,SMX,SK,SL,QVL)

= QMXSUM(PQMCON(QVK,SMA,SK,SL,QVL) ,

PQMCON(QVK,SMA,SCOMP(SK),SCOMP(SL),QVL))

and is expressible as the Boolean sum of two 3x3 matrix connectives.

PQ(ii): PQBNPT for the formulae (5) to (7)

PQBNPT(QVA,QVK,SMX,SK,SL,QVL,QVB) = BC

where

QVA = $\underset{\sim}{a}$, QVB = $\underset{\sim}{b}$, QVK = $(\underset{\sim}{q_z}x)$, QVL = $(\underset{\sim}{q_z}y)$

For SMX = SMA,SMO,SMI,SME

BC = QBINPT(QVA,QMXP,QVB)

where

QMXP = PQMCON(QVK,SMX,SK,SL,QVL)

The calculation of the SNS truth value PQSTV, of a binary relation in PQLOG, is also readily obtained in terms of the above subroutine PQBNPT, as follows.

PQ(iii): PQSTV: - SNS truth value of a binary relation in QL-1B

The relevant subroutine is

PQSTV(QVA,QVK,SMX,SK,SL,QVL,QVB) = SVC

SVC(1) = QBINPT(QVA,QMXP,QVB)

SVC(2) = QB1NPT(QVA,QCOMP(QMXP),QVB)

where

QMXP = PQMCON(QVK,SMX,SK,SL,QVL)

PQ(iv): PQUNPT : Unary product in QL-1B

The unary relation in PQLOG leads to a quantifier state as output as described in (8a,b,c), and here also, the function is evaluated by using the equivalent formula QUNPDT in QLOGIC as follows:

PQUNPT(QVA,QVK,SMX,SK,SL,QVL) = QVB

This is evaluated as

QVB = QUNPDT(QVA,QMXP)

For SMX = SMA,SMO,SMI,SME

QMXP = PQMCON(QVK,SMX,SK,SL,QVL)

Note that for SME, we use the same definition of the equivalent QL-2 matrix as adopted in PQ(i), and described in the algebra mentioned in Section 5(b).

In this manner, all elementary statements in QL-1B are convertible into the QL-2 form, either as a unary product, or a binary product, and the whole structure of QL-2 can be used for working them out. It is to be noted that for obtaining the SNS truth value PQSTV in QL-1B, the complementary relation is calculated in the equivalent QL-2 form. The need and consequences of this procedure will be explained in the report to be prepared, dealing with the improved algebraic theory of QL-1B.

MATLOG Program in FORTRAN

Part III- Program for GLOGIC and the theory of relations in general

G.N. Ramachandran
INSA Albert Einstein Professor

and

T.A. Thanaraj*
Visiting Fellow

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

*(Permanent Address: Centre for Cellular and Molecular Biology
Hyderabad 500 007)

MR - 62, AUGUST 1987

Part III: Program for GLOGIC and the theory of relations in general

G.N. Ramachandran
INSA Albert Einstein Professor

and

T.A. Thanaraj*
Visiting Fellow

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

*(Permanent Address: Centre for Cellular and Molecular
Biology, Hyderabad 500 007.)

, August 1987

CONTENTS

Page No.

1 Introduction 1

2 GLOGIC 2

(a) Additional subroutines for basic Boolean scalars 2

(b) GLOGIC functions and subroutines 2

3 Subroutines for GCLOG 6

4 Applications in examples 9

(a) Illustration of unary relations 11

(b) Illustrations of binary relations in GCLOG 11

5 Operations for the combination of terms and relations 15

(a) Boolean operations on matrices 15

(b) Combination of two matrix relations 23

(c) Union of two matrix relations 26

(d) Expanded product of two matrix relations 27

(e) Tensor relation $\underset{\sim}{R}(i,j,k)$ applied in three ways 28

6 Generalized clausal relations 28

7 Aristotle's Syllogisms in GLOGIC 32

8 Mixed statements containing quantified and unquantified terms *via* GLOGIC 36

9 Comment on the combination of two relations in general 38

(a) Unary relations and Boolean truth values of binary relations 39

(b) Conditions for conjunction identity 41

(c) Extension to GLOGIC 45

(d) SNS truth values 47

Part III: MATLOG Program for GLOGIC and the theory of relations in general

1. Introduction

The theory of relations in Boolean algebra employing m x n Boolean matrices was developed first in MR-42 and has thereafter been applied in various ways. Examples of its application to data processing and graph theory were developed in ALOG-31 to 34 and the application of general Boolean matrices of this type to answer various questions in multivalued logic considered in MR-46A.

In this report, we shall consider essentially the FORTRAN programing of the relevant equations. They follow essentially the same pattern as for SNSLOG and QLOGIC, and deal with m x n matrices, instead of 2x2 and 3x3 matrices for SNS and QL-2. Since the numbers m and n have to be left open, to be chosen suitably for each problem, the structure of the programs is slightly different. However, the subroutines that are listed follow closely the sequence that was adopted for SNS and QL-2. We shall call this as General Logic (GLOGIC) and indicate all vectors and matrices and other operators in this logic by the first letter G for their symbols. The basic subroutines are described in Sections 2 and 3 and thereafter various applications of these are considered.

2. GLOGIC

In the general case, vectors and matrices will have the format GVA (MI) and GMZ (MI,MJ). The former will have components GVA(I), I = 1 , MI, and the latter GMZ (I,J) I = 1, MI, J = 1 , MJ.

(a) Additional subroutines for basic Boolean scalars

We define three functions "Boolean multiple sum" (BMSUM) "Boolean multiple product" (BMPDT) and "Boolean multiple Equivalence" (BMEQU) as follows

```
            BMSUM(BA(I), MI) = BC
stands for   BMSUM = 0,  DO I = 1, MI
             BMSUM = BSUM(BMSUM, BA(I))
```

The functions BMPDT and BMEQU are given by similar equations as follows

```
             BMPDT = 0, DO  I = 1, MI
             BMPDT = BPDT(BMPDT,BA(I))

             BMEQU = 1,  DO  I = 1, MI
             BMEQU = BEQU(BMEQU, BEQU(BA(I),BA(1)))
```

(b) GLOGIC functions and subroutines

i) Boolean sum of two vectors : $\underset{\approx}{a} \oplus \underset{\approx}{b} = \underset{\approx}{c}$

GUNION (GVA,GVB,MI) = GVC

Check that both GVA and GVB have MI components

BSUM (GVA(I), GVB(I)) = GVC(I) I = 1, MI

(ii) Boolean product of two vectors : $\underset{\approx}{a} \otimes \underset{\approx}{b} = \underset{\approx}{c}$

```
GVIDYA (GVA,GVB,MI) = GVC
```

Check that both GVA and GVB have MI components

```
BPDT (GVA(I),GVB(I)) = GVC(I)   I = 1, MI
```

(iii) Boolean complement of a vector : $\overline{\underset{\approx}{a}}^{c} = \underset{\approx}{b}$

```
GCOMP (GVA,MI) = GVB

BCMP (GVA(I)) = GVB(I)    I = 1,  MI
```

(iv) Boolean sum of two matrices : $\underset{\approx}{P} \oplus \underset{\approx}{Q} = \underset{\approx}{R}$

```
GMXSUM (GMP,GMQ,MI,MJ) = GMR
```

Check that both GMP , GMQ have MI,MJ components

```
BSUM (GMP(I,J), GMQ(I,J)) = GMR(I,J)
                     I = 1 , MI
                     J = 1 , MJ
```

(v) Boolean product of two matrices : $\underset{\approx}{P} \otimes \underset{\approx}{Q} = \underset{\approx}{R}$

```
GMXPDT (GMP,GMQ,MI,MJ) = GMR

BPDT (GMP(I,J), GMQ(I,J)) = GMR(I,J)

                     I = 1 , MI
                     J = 1 , MJ
```

(vi) Boolean complement of a matrix operator : $\underset{\approx}{P}^{c} = \underset{\approx}{Q}$

```
GMXCMP (GMP,MI,MJ) = GMQ
  BCMP (GMP(I,J)) = GMQ(I,J)    I = 1 , MI
                                J = 1 , MJ
```

(vii) Scalar product of two vectors $a_1 \otimes b_1 \oplus a_2 \otimes b_2 \oplus \ldots a_m \otimes b_m = c$

```
GSCPDT (GVA,GVB,MI) = BC

BMSUM(BPDT (GVA(I),GVB(I)),MI) = BC
```

viii) Unary product of a vector with a matrix

$$\bigoplus_{i} (a_i \otimes z_{ij}) = b_j$$

```
GUNPDT (GVA,GMZ,MI,MJ) = GVB
Input  GVA(MI), GMZ(MI,MJ)

Output  GVB(MJ)

BMSUM (BPDT (GVA(I),GMA(I,J), MI)),MI) = GVB(J)

                                 J = 1, MJ
```

ix) Binary product of a matrix with two vectors

(Boolean truth value) $\langle \underset{\approx}{a} | \underset{\approx}{Z} | \underset{\approx}{b} \rangle' = c$

```
GBTV (GVA,GMZ,GVB,MI,MJ) = BC

GVAP = GUNPDT (GVA,GMZ,MI,MJ)

  BC = GSCPDT (GVAP,GVB,MJ)
```

x) Matrix product of two matrices : $|\underset{\approx}{P}|\underset{\approx}{Q}| = |\underset{\approx}{R}|$

```
GMATPT (GMP,GMQ,MI,MJ,MK) = GMR

BMSUM (BPDT(GMP(I,J),GMQ(J,K)),MJ) = GMR(I,K)

     I = 1 to MI ,    K = 1 to  MK
```

xi) Direct product of two vectors : $\underset{\approx}{a} \times \underset{\approx}{b} = \underset{\approx}{Z}$

```
GDIRPT(GVA,MI,GVB,MJ) = GMZ

GMZ(I,J) = BPDT(GVA(I),GVB(J))

              I = 1, MI
              J = 1, MJ
```

xii) Direct sum of two vectors : $\underset{\approx}{a} + \underset{\approx}{b} = \underset{\approx}{Z}$

```
GDIRSM(GVA,MI,GVB,MJ) = GMZ
GMZ(I,J) = BSUM(GVA(I),GVB(J))

              I = 1, MI
              J = 1, MJ
```

xiii) Direct equivalence of two vectors: $\underset{\sim}{a} \equiv \underset{\sim}{b} = \underset{\approx}{Z}$

GDIREQ(GVA,MI,GVB,MJ) = GMZ

GMZ(I,J) = BEQU(GVA(I),GVB(J))

I = 1, MI
J = 1, MJ

xiv) Scalar vector direct product : $a \times \underset{\approx}{b} = \underset{\approx}{c}$

GSVDPT(BA,GVB,MI) = GVC

GVC(I) = BPDT(BA,GVB(I))

xv) Scalar vector direct sum : $a + \underset{\approx}{b} = \underset{\approx}{c}$

GSVDSM(BA,GVB,MI) = GVC

GVC(I) = BSUM(BA,GVB(I)) I = 1, MI

xvi) Transpose of a matrix : $\underset{\approx}{P}^{t} = \underset{\approx}{Q}$

GTRANS(GMP,MI,MJ) = GMQ

GMQ(I,J) = GMP(J,I)

I = 1, MI
J = 1, MJ

xvii) ~~Matrix transformation operators~~ $|\underset{\approx}{Z}|^{o} = |\underline{Z}|' = {}^{e}, {}^{c}, {}^{t}, {}^{ct}$

Since these operators occur frequently in general classical logic involving multiple element sets, they are given short names in MATLOG, namely GE, GC, GT, GTC. They are defined as follows:

a) GE(GMZ,MI,MJ) = GMZ1
GMZ1(I,J) = GMZ(I,J)

b) GC(GMZ,MI,MJ) = GMZ2
GMZ2,MI,MJ = GMXCMP(GMZ,MI,MJ)

c) GT(GMZ,MI,MJ) = GMZ3
GMZ3,MJ,MI = GTRANS(GMZ,MI,MJ)

d) GTC(GMZ,MI,MJ) = GMZ4
GMZ4,MJ,MI = GMXCMP(GTRANS(GMZ,MI,MJ)

3. Subroutines for GCLOG

The programs for General Classical Logic, (GCLOG) are somewhat different from SNSLOG and are indicated by general logical relational matrices $\underset{\sim}{A}(\underset{\sim}{k},\underset{\approx}{\ell})$, $\underset{\sim}{O}(\underset{\sim}{k},\underset{\approx}{\ell})$, etc. The subroutines suggested are as follows.

i) Definition of relational matrices: $\underset{\sim}{Z}(\underset{\sim}{k},\underset{\approx}{\ell})$, $Z = A, O, E, I$

GMX, GVK, GVL, X = A, O, E, I

These are analogous to (xvii) SMZ, SK,SL of SNSLOG

```
GMA,GVK,GVL = GLIRPT(GVK,GVL,MK,ML)

GMO,GVK,GVL = GDIRSM(GVK,GVL,MK,ML)

GME,GVK,GVL = GDIREQ(GVK,GVL,MK,ML);

GMI,GVK,GVL = GDIRSM(GCOMP(GVK),GVL,MK,ML)
```

ii) Relative (SNS) truth value of one term $\underset{\approx}{a}$ for another $\underset{\approx}{b}$:

```
GRELTV(GVA,GVB,MI) = SVC
```

$t(\underset{\sim}{a}\,|\,\underset{\sim}{b}) = c$

This is the general case of (xviii) SRELTV

```
SVC(1) = GSCPDT(GVA,GVB)

SVC(2) = GSCPDT(GVA,GCOMP(GVB))
```

iii) SNS truth value of the relation $\underset{\approx}{a}\,\underset{\approx}{Z}\,\underset{\approx}{b}$

$$\underset{=}{t}(\underset{\approx}{a}\,\underset{\approx}{Z}\,\underset{\approx}{b}) = \underset{=}{c} = (c_1,c_2)$$

where $c_1 = \langle \underset{\sim}{a} \,|\, \underset{\sim}{Z} \,|\, \underset{\sim}{b} \rangle$, $c_2 = \langle \underset{\sim}{a} \,|\, \underset{\sim}{Z}^{c} \,|\, \underset{\sim}{b} \rangle$

```
GSTV(GVA,GML,GVB,MI,MJ) = SVC
```

This is the general case of (xix) SSTV

```
        SVC(1) = GBTV(GVA,GMZ,GVB,MI,MJ)

        SVC(2) = GBTV(GVA,GMXCMP(GMZ),GVB,MI,MJ)
```

iv) SNS truth value for $\underset{\approx}{Z}$ = Z($\underset{\approx}{k}$, $\underset{\approx}{\ell}$), Z = A, O, E, I via relative truth value

```
        GSTV2(GVA,GMX,GVK,GVL,GVB,MI,MJ) = SVC

    If GMX = GMA

    SVC(1) = BPDT(GSCPDT(GVA,GVK,MI), GSCPDT(GVB,GVL,MJ))

    SVC(2) = BSUM(GSCPDT(GVA,GCOMP(GVK),MI),GSCPDT(GVB,GCOMP(GVL),MJ))

    If GMX = GMO

    SVC(1) = BSUM(GSCPDT(GVA,GVK,MI),GSCPDT(GVB,GVL,MJ))

    SVC(2) = BPDT(GSCPDT(GVA,GCOMP(GVK),MI),GSCPDT(GVB,GCOMP(GVL),
                                                              MJ))

    If GMX = GME

        GVKP = GCOMP(GVK)

        GVLP = GCOMP(GVL)

        GVCA = GSTV2(GVA,GMA,GVK,GVL,GVB,MI,MJ)

        GVCB = GSTV2(GVA,GMA,GVKP,GVLP,GVB,MI,MJ)

        SVC(1) = BSUM(GVCA(1), GVCB(1))

        SVC(2) = BPDT(GVCA(2), GVCB(2))

    If GMX = GMI

        GVKP = GCOMP(GVK)

        GMI  = GMO,GVLP,GVL
```

v) Output of unary relation for $\underset{\approx}{Z}$ ($\underset{\approx}{k}$, $\underset{\approx}{\ell}$) using the relative truth value of the input

```
GUNPT2(GVA,GMX,GVK,GVL,MI,MJ) = GVB

If GMX = GMA

    BAP = GSCPDT(GVA,GVK,MI)

    GVB = GSVDPT(BAP,GVL,MJ)

If GMX = GMO

    BAP = GSCPDT(GVA,GVK,MI)

    GVB = GSVDSM(BAP,GVL,MJ)

If GMX = GME

   GVKP = GCOMP(GVK), GVLP = GCOMP(GVL)

   GVBA = GUNPT2(GVA,GMA,GVK,GVL,MI,MJ)

   GVBB = GUNPT2(GVA,GMA,GVKP,GVLP,MI,MJ)

   GVB  = GUNION(GVBA,GVBB,MJ)

If GMX = GMI

    BAP = GSCPDT(GVA,GCOMP(GVK),MI)

    GVB = GSVDSM(BAP,GVL,MJ)
```

vi) Binary operator "agree" ($\underset{\approx}{G}$) for the general case

```
GMBG(GVA,GVB,MI) = SVC

BCP(I) = BEQU(GVA(I),GVB(I))  I = 1, MI

SVC(1) = BMPDT(BCP,MI)

SVC(2) = BCMP(SVC(1))
```

vii) Matrix-Boolean unary operators $\underset{\sim}{\underline{E}}$, $\underset{\sim}{\underline{N}}$, $\underset{\sim}{\underline{M}}$, $\underset{\sim}{\underline{L}}$

```
GEQU(GVA,MI) = GVB
GVB(I) = GVA(I)    I = 1, MI

GNOT(GVA,MI) = GVB
GVB(I) = GVA(MI - I + 1)    I = 1, MI

GCOMP(GVA,MI) = GVB
GVB(I) = BCMP(GVA(I))       I = 1, MI

GELL(GVA,MI) = GVB
GVB(I) = BCMP(GVA(MI - I + 1)    I = 1, MI
```

4. Applications in examples

We shall first give illustrations of the unary and binary relations involving a general matrix connective R_{ij} (i = 1 to m, j = 1 to n), taking for this purpose illustrations from Tables 2 and 4 of IR-42. All of them are based on the student-professor" relation and its reverse relation"professor-student" described in Table 1 of that reference. For ready reference, the four matrices $\underset{\approx}{P}$, $\underset{\approx}{P}^{c}$, $\underset{\approx}{P}^{t}$, $\underset{\approx}{P}^{tc}$ which are derivable from the relation $\underset{\approx}{P}$, standing for "professor of", are explicitly described in Table 1 below. They make use of the four matrix-Boolean operators GE, GT, GC, GTC as applied to the matrix $\underset{\approx}{P}$.

Table 1. The four matrices* $\underset{\approx}{P}$, $\underset{\approx}{P}^c$, $\underset{\approx}{P}^t$, $\underset{\approx}{P}^{tc}$ employed for the relations† in the student-professor problem

$$\underset{\approx}{s} \xrightarrow{\underset{\approx}{P}} \underset{\approx}{p}$$

$$|\underset{\approx}{P}| = \mathrm{GM1}(5,4) = \begin{pmatrix} 1&0&1&0\\ 0&1&1&0\\ 1&0&0&1\\ 0&0&0&1\\ 1&1&1&0 \end{pmatrix}$$
$$= \mathrm{GE}(\mathrm{GM1}(5,4))$$

$$\underset{\approx}{p} \xrightarrow{\underset{\approx}{S}} \underset{\approx}{s}$$

$$;\ |\underset{\approx}{S}| = |\underset{\approx}{P}^t| = \mathrm{GM3}(4,5) = \begin{pmatrix} 1&0&1&0&1\\ 0&1&0&0&1\\ 1&1&0&0&1\\ 0&0&1&1&0 \end{pmatrix}$$
$$= \mathrm{GT}(\mathrm{GM1}(5,4))$$

$$\underset{\approx}{s} \xrightarrow{\underset{\approx}{P}^c} \underset{\approx}{p}$$

$$\underset{\approx}{P}^c = \mathrm{GM2}(5,4) = \begin{pmatrix} 0&1&0&1\\ 1&0&0&1\\ 0&1&1&0\\ 1&1&1&0\\ 0&0&0&1 \end{pmatrix}$$
$$= \mathrm{GC}(\mathrm{GM1}(5,4))$$

$$\underset{\approx}{p} \xrightarrow{\underset{\approx}{S}^c} \underset{\approx}{s}$$

$$;\ \underset{\approx}{S}^c = \underset{\approx}{P}^{tc} = \mathrm{GM4}(4,5) = \begin{pmatrix} 0&1&0&1&0\\ 1&0&1&1&0\\ 0&0&1&1&0\\ 1&1&0&0&1 \end{pmatrix}$$
$$= \mathrm{GTC}(\mathrm{GM1}(5,4))$$

*These are typical examples, and the same pattern holds for any matrix relation $|R|$, and four related matrices $|R|$, $|R^c|$, $|R^t|$, $|R^{tc}|$ are definable.

†The single arrow ($\longrightarrow$) is used to indicate a relation in the general theory of relations, which is to be distinguished from the double arrow ($\Longrightarrow$) for implication, and the arrow with a vertical line ($\longmapsto$) for "leads to ".

(a) Illustration of unary relations

A number of chosen examples of these are given in Table 2. The querries are self-explanatory and the relevant vector-matrix equation that gives the solution to the problem in each case is given on the r.h.s. The FORTRAN statement in MATLOG is given below the query, and the input and output are given below that. In these simple problems, they can be readily verified to be true by inspection. However, the FORTRAN statements indicate that a whole variety of related questions that can be asked, can be expressed in terms of the basic functions which we have defined in FORTRAN for GLOGIC.

(b) Illustrations of binary relations in GCLOG

These are summarized in Table 3 which is given below, and since they are taken from MR-42, no details are given and only the corresponding Boolean algebraic vector-matrix formula is given in each case. It can be readily verified that our library of FORTRAN statements can readily translate them all into a format suitable for application with our MATLOG subroutines It is also evident that a whole variety of analogous problems can be solved by combining these formulae of MATLOG in a suitable manner.

Table 2. Some examples of unary relations in GCLOG*

*1. Who are all the professors (p') who teach the students in $\underline{\underline{s}}$' ? $\langle s \,|\, P \,| = \langle p \,|$

GUNPDT(GVS,GM1,MI,MJ) = GVP

GVS = (1 1 0 0 0), GVP = (1 1 1 0)

4. Who are all the non-students($\underline{\underline{s}}$") of some professors in $\underline{\underline{p}}$' ? $\langle p \,|\, P^{tc} \,| = \langle s \,|$

GUNPDT(GVP,GM4,MJ,MI) = GVS

GVP = (1 1 0 0) , GVS = (1 1 1 1 0)

5. Who are all the students ($\underline{\underline{s}}$") attending classes taken by professors that teach $\underline{s}$'? $\langle s1 \,|\, P \,|\, P^{t} \,| = \langle s2 \,|$

GMATPT(GM1,GM3,MI,MJ,MI) = GM6

GUNPDT(GVS1,GM6,MI,MJ) = GVS2

GVS1 = (1 1 0 0 0) , GVS2 = (1 1 1 0 1)

7. Which professors ($\underline{\underline{p}}$") in p' teach some student in $\underline{s}$'? $\langle s \,|\, P \,| \otimes \langle p1 \,| = \langle p2 \,|$

GUNPDT(GVS1,GM1,MI,MJ) = GVP

GVIDYA(GVP,GVP1) = GVP2

GVS1 = (1 1 0 0 0), GVP1 = (0 1 0 1), GVP2 = (0 1 0 0)

10. Which are the students ($\underline{\underline{s}}$') that take the lectures of p', and do not attend the classes of $\underline{\underline{p}}$"? $\langle p1 \,|\, P^{t} \,| \otimes \langle p2 \,|\, P^{tc} \,| = \langle s \,|$

GUNPDT(GVP1,GM3,MJ,MI) = GVS1
GUNPDT(GVP2,GM4,MJ,MI) = GVS2
GVIDYA(GVS1,GVS2,MI) = GVS
GVP1 = (1 1 0 0), GVP2(0 1 1 0), GVS(1 0 1 0 0)

* From Table 2, MR-42.

12. Which students (s') are taught by professors that do not belong to the subset p'? $\langle p^c | P^t | = \langle s |$

GUNPDT(GCOMP(GVP),GM3) = GVS

GVP = (0 1 0 1), GVS = (1 1 1 0 1)

Table 3. Some examples of binary checking formulae in the BVM formulation of general classical relations*

*1 . Check if at least one of s' is related by P to one of p' . $\langle s | P | p \rangle = 1$

GBTV(GVS,~~GM1~~,GVP,MI,MJ) = BTV ; BEQU(BTV,B1) = BC

GVS = (1 1 0 0 0), GVP = (0 1 1 0), BTV = 1, BC = 1 (Yes)

3 . Check if at least one of p' is non-related by S to one of s'. $\langle p | S^c | s \rangle = 1$

GBTV(GVP,GM4,GVS,MJ,MI) = BTV, BEQ(BTV,B1) = BC

GVP = (0 1 1 0), GVS = (0 0 1 1 0), BTV = 1, BC = 1 (Yes).

4 . Check if all of p' are non-related by S to any of the students in s' . $\langle p | S | s \rangle = 0$

GBTV(GVP,GM3,GVS,MJ,MI) = BTV, BEQ(BTV,B2) = BC

GVP = (0 1 1 0), GVS = (0 0 1 1 0), BTV = 0, BC = 1 (Yes)
All are non-related.

* From Table 4 of MR-42

Table 3 . Contd.

6 . Check if the professors p' teaching $\underline{\underline{s}}'$ have any members in common with those $\underline{\underline{p}}''$ teaching $\underline{\underline{s}}''$.

$$\langle s1 | P | P^t | s2 \rangle = 1$$

```
GMATPT(GM1,GM3,MI,MJ,MI) = GM5

GBTV(GVS1,GM5,GVS2,MI,MI) = BTV,  BEQU(BTV,B1) = BC

GVS1 = (1 0 1 0 0), GVS2 = (01 0 0 1), BTV = 1 , BC = 1 (Yes
```

7 . If the answer to Problem 6 is "yes", list the set of professors ($\underline{\underline{p}}_o$) common to both.

$$\langle s1 | P | = \langle p1 | ,$$
$$\langle s2 | P | = \langle p2 | ,$$
$$\langle p1 | \otimes \langle p2 | = \langle p_o |$$

```
GUNPDT(GVS1,GM1,MI,MJ) = GVP1

GUNPDT(GVS2,GM1,MI,MJ) = GVP2

GVIDYA(GVP1,GVP2,MJ) = GVPO

GVP1 = (1 0 1 1), GVP2 = (1 1 1 0), GVPO = (1 0 1 0)
```

8 . Check if $\underline{\underline{s}}'$ contains all the students studying under the professors ($\underline{\underline{p}}'$) who teach any of them.

$$| P | S | = | K | ,$$
$$\langle s | K | s^c \rangle = 0$$

```
GMATPT(GM1,GM3,MI,MJ,MI) = GMK

GBTV(GVS1,GMK,GCOMP(GVS1),MI,MJ) = BTV, BEQU(BTV,B2) =BC

GVS1 = (1 0 0 0 1 ) , BTV = 1, BC = 0  (No).
```

9. If "No", list the set of professors ($\underline{\underline{p}}'$) and also the full list of students ($\underline{\underline{s}}_o$) studying under them.

$$\langle s1 | P | = \langle p1 |$$
$$\langle p1 | S | = \langle s_o |$$

```
GUNPDT(GVS1,GM1,MI,MJ) = GVP1

GUNPDT(GVP1,GM3,MJ,MI) = GVSO

GVP1 = (1 1 1 0), GVSO = (1 1 1 0 1)
```

Before considering the particular case of GCLOG which is relevant for clausal relations, and which employ the subroutines involving GMX,GVK,GVL described above, we shall make some comments regarding a general problem, namely that of the application of Boolean connective operators $\oplus$, $\otimes$, ^{c} to matrix connectives, in addition to vectors representing terms. This is given in the next Section 5 and some examples are given, after which the special case of clausal relations, and their generalization into four types as explained in MR-57, will be taken up.

5. Operations for the combination of terms and relations

(a) Boolean operations on matrices

Quite apart from the vector-matrix products which are relevant for the "matrix" type connective $|\underset{\sim}{P}|$ and the related matrices $|\underset{\sim}{P}^c|$, $|\underset{\sim}{S}|$ and $|\underset{\sim}{S}^c|$, which were considered above, we have also made use of "Boolean" connective operators denoted by the symbols $\oplus$, $\otimes$ and ^{c} in some of the problems considered in Tables 2 and 3. However, in these applications, these Boolean connectives follow the same definitions as in SMS and GL-2, namely that they correspond to the combination of information regarding the <u>same entity</u>, but coming from different sources. When applied to "terms" represented by vectors in BVMF, the two operators, $\oplus$, standing for "union" ($\underset{\sim}{\underline{U}}$) , and $\otimes$, standing for "vidya" ($\underset{\sim}{\underline{V}}$) , correspond to the union and intersection of the sets A1 and A2, whose members are represented by the 1's in the corresponding vectors $\underset{\sim}{\underline{a1}}$ and $\underset{\sim}{\underline{a2}}$ respectively. Thus, we obtain the resultant set A, represented by $\underset{\sim}{\underline{a}}$, given by the vector equations

$$\underset{\sim}{\underline{a1}} \oplus \underset{\sim}{\underline{a2}} = \underset{\sim}{\underline{a}} \, , \quad \underset{\sim}{\underline{a1}} \otimes \underset{\sim}{\underline{a2}} = \underset{\sim}{\underline{a}} \qquad (1)$$

in these two cases.

Similarly, it is also possible to apply the operations of "union" and "vidya" to "relations" represented by matrices. We have already employed this intuitively in SNS and QLOG (e.g. in the definition of the equivalence operator), but this requires more specific definition and description for the general case. Thus, the operations $\oplus$ and $\otimes$ are required in situations involving the simultaneous existence of two relations $\underset{\sim}{\underline{R1}}$ and $\underset{\sim}{\underline{R2}}$ as applied to the same state vector $\underset{\sim}{\underline{a}}$, leading to two outputs $\underset{\sim}{\underline{b1}}$ and $\underset{\sim}{\underline{b2}}$ respectively for $\underset{\sim}{\underline{b}}$. The question then arises : "What is the effective resultant output vector $\underset{\sim}{\underline{b}}$ if the two relations are both applicable to a problem, either conjunctively, or disjunctively, as the case may be?" As we shall see below, this can be done by applying the Boolean sum, or product, operation to the corresponding elements of the connective matrices $\underset{\sim}{\underline{\mathrm{R1}}}$ and $\underset{\sim}{\underline{\mathrm{R2}}}$ respectively. Thus, for two relations in conjunction

$$\underset{\sim}{\underline{a}}\,\underset{\sim}{\underline{\mathrm{R1}}} = \underset{\sim}{\underline{\mathrm{b1}}}\,,\quad \underset{\sim}{\underline{a}}\,\underset{\sim}{\underline{\mathrm{R2}}} = \underset{\sim}{\underline{\mathrm{b2}}} \quad\longmapsto\quad \underset{\sim}{\underline{a}}\,(\underset{\sim}{\underline{\mathrm{R1}}} \otimes \underset{\sim}{\underline{\mathrm{R2}}}) = \underset{\sim}{\underline{b}}\,, \tag{2a}$$

and for two relations in disjunction, they lead to

$$\underset{\sim}{\underline{a}}\,(\underset{\sim}{\underline{\mathrm{R1}}} \oplus \underset{\sim}{\underline{\mathrm{R2}}}) = \underset{\sim}{\underline{b}} \quad \text{—} \tag{2b}$$

We have already defined the Boolean operations of sum, and product, as applied to matrices, in the subroutines GMXSUM and GMXPDT, and they are definable as in (3a) and (3b) below :

Boolean product: $\mathrm{R1}_{i,j} \otimes \mathrm{R2}_{i,j} = \mathrm{R}_{i,j}\,,$ (3a)

Boolean sum : $\mathrm{R1}_{i,j} \oplus \mathrm{R2}_{i,j} = \mathrm{R}_{i,j}\,,$ (3b)

for each (i,j), $i = 1$ to m , $j = 1$ to n (3c)

The logical reason as to why the Boolean operation has to be applied between the components of the matrix elements representing the connectives, rather than vector components of the two outputs $\underset{\sim}{\underline{\mathrm{b1}}}$ and $\underset{\sim}{\underline{\mathrm{b2}}}$, in order to obtain $\underset{\sim}{\underline{b}}$

can be given as follows. A general matrix relation $\underset{\sim}{\underline{a}}\ \underset{\sim}{\underline{R}} = \underset{\sim}{\underline{b}}$ in GLOGIC really stands for the set of mxn relations $a_i\ R_{ij} = b_j$; $i = 1$ to m, $j = 1$ to n, in which each individual element a_i of the input is either related, or non-related, to each individual element b_j of the output, according as $R_{ij} = 1$, or 0. Therefore, when two relations $R1_{i,j}$ and $R2_{i,j}$ are coexistent, then each equation $a_i\ R_{ij} = b_j$ leading to b_j from a_i has to be treated separately. Effectively, we thus have mxn equations, for all pairs (i,j) with $i = 1$ to m, $j = 1$ to n, as in (4a) and (4b), corresponding to (3a) and (3b) respectively.

$$b_j = a_i\ R1_{i,j} \oplus a_i\ R2_{i,j} = a_i(R1_{i,j} \oplus R2_{i,j}) \qquad (4a)$$

$$b_j = a_i\ R1_{i,j} \otimes a_i\ R2_{i,j} = a_i(R1_{i,j} \otimes R2_{i,j}) \qquad (4b)$$

These evidently lead to the matrix equations (2a) and (2b), respectively, in BVMF, for the general case.

To complete the discussion, we shall indicate that the Boolean complementation operator, applied to a <u>relation</u>, leads from the equation $\underset{\sim}{\underline{a}}\ \underset{\sim}{\underline{R}} = \underset{\sim}{\underline{b}}$ to $\underset{\sim}{\underline{a}}\ \underset{\sim}{\underline{R}}^c = \underset{\sim}{\underline{b}}'$, where the elements of R_{ij} are negated, as in

$$\underset{\sim}{\underline{R}}^c_{i,j} = (\underset{\sim}{\underline{R}}_{i,j})^c \qquad (5)$$

It should be noted that $\underset{\sim}{b}' \neq \underset{\sim}{b}^c$ in general, and the two operations, of complementing a relation, and of complementing the output, represented by the MATLOG functions GMXCMP and GCOMP respectively, are entirely different from one another.

If these distinctions regarding the Boolean operations as applied to terms and relations are borne in mind and properly applied, then practically any problem involving relations in GLOGIC can be treated, requiring only one or more of the operations that have been described above in MATLOG. We shall give a few illustrative examples below.

Table 4. Illustrations of Boolean operations on connectives in GCLOG

Inputs to the problem

Two sets represented by the vectors $\underset{\sim}{a}$ and $\underset{\sim}{b}$ are related by two different properties, namely

$\underset{\sim}{P}$: b_j is the professor of a_i

$\underset{\sim}{E}$: b_j is the examiner of a_i

$$|\underset{\sim}{P}| = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}, \quad |\underset{\sim}{E}| = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

A : Conjunction of two relations

Find those members of the set, represented by $\underset{\sim}{b3}$, who are both professors of, as well as the examiner of the set represented by vector $\underset{\sim}{a} = (0\ 1\ 1\ 0\ 0)$.

The relevant matrix is

$$|\underset{\sim}{R1}| = |\underset{\sim}{P}| \otimes |\underset{\sim}{E}| = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

$$\text{and } \langle \underset{\sim}{b3}| = \langle \underset{\sim}{a} | \underset{\sim}{R1}| = (0\quad 0\quad 0\quad 1)$$

Note that the relations $\underset{\sim}{P}$ and $\underset{\sim}{E}$ individually lead to the sets $\underset{\sim}{b1}$ and $\underset{\sim}{b2}$ given by the vectors

$$\langle \underset{\sim}{b1}| = (1\quad 1\quad 0\ 1),\ \langle \underset{\sim}{b2}| = (1\quad 1\quad 1\quad 1)$$

and

$$\langle \underset{\sim}{b3}| \neq \langle \underset{\sim}{b1}| \otimes \langle \underset{\sim}{b2}| = \langle \underset{\sim}{b4}| = (1\ 1\ 0\ 1)$$

B : Disjunction of two relations

Find the representive vector $\underset{\sim}{b5}$ corresponding to the members of the set $\underset{\sim}{b}$, who are either professors of at least one member of $\underset{\sim}{a1}$, or are the examiners of on? member of $\underset{\sim}{a1}$.

Clearly, the relevant matrix is

$$|\underset{\sim}{\underline{R2}}| = |\underset{\sim}{\underline{P}}| \bigoplus |\underset{\sim}{\underline{E}}| = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

and $\langle \underset{\sim}{\underline{b5}}| = \langle \underset{\sim}{\underline{a}}\, |\underset{\sim}{\underline{R2}}| = (1 \quad 1 \quad 1 \quad 1)$

Note that, in this case, $\langle \underset{\sim}{\underline{b5}}| = \langle \underset{\sim}{\underline{b1}}| \bigoplus \langle \underset{\sim}{\underline{b2}}|$, which can be proved quite generally for the Boolean sum of two relations.

If we represent $\underset{\sim}{\underline{P}}$ and $\underset{\sim}{\underline{E}}$ by GMP and GMC, and $\underset{\sim}{\underline{a}}$ and $\underset{\sim}{\underline{b}}$ by GVA and GVB, then the FORTRAN statement for calculating $\underset{\sim}{\underline{b3}}$ in MATLOG is

```
GVB3 = GUNPDT(GVA,GMXPDT(GMP,GME),MI,MJ))
```

while the corresponding statement for calculating $\underset{\sim}{\underline{b4}}$ is

```
GVB4 = GVIDYA(GUNPDT(GVA,GMP), GUNPDT(GVA,GME), MI,MJ)
```

The MATLOG formula for calculating $\underset{\sim}{\underline{b5}}$ is, similarly,

```
GVB5 = GUNPDT(GVA,GMXSUM(GMP,GME),MI,MJ)
```

C. Complementation of a relation

The Boolean complement has already been discussed in Table 2, both for a matrix relation, as well as for a vector term. In particular, for our problem, the complement of $\underset{\sim}{\underline{P}}$ leads to the BVMF equation

$$\langle \underset{\sim}{\underline{a}}\, | \underset{\sim}{\underline{P}}^{c}| = \langle \underset{\sim}{\underline{b6}}|$$

giving the set of $\underset{\sim}{\underline{b}}$'s who do not take classes for some of the $\underset{\sim}{\underline{a}}$'s in the set under consideration.

To illustrate the distinction between $\langle \underset{\sim}{\underline{a}} | \underset{\sim}{\underline{P}}^{c} | = \langle \underset{\sim}{\underline{b6}} |$ and $(\langle a | \underset{\sim}{\underline{P}} |)^{c} = \langle \underset{\sim}{\underline{b7}} |$, we note that they are programable in MATLOG by the statements

```
GVB6 = GUNPDT(GVA,GMXCMP(GMP),MI,MJ)
```

and

```
GVB7 = GCOMP(GUNPDT(GVA,GMP), MI, MJ))
```

Taking the example given above, we obtain

$$\langle \underset{\sim}{\underline{b6}} | = (0\ 1\ 1\ 0\ 0) \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = (1\ 1\ 1\ 1)$$

$$\langle \underset{\sim}{\underline{b7}} | = \langle \underset{\sim}{\underline{b1}} |^{c} = (1\ 1\ 0\ 1)^{c} = (0\ 0\ 1\ 0)$$

The difference between $\underset{\sim}{\underline{b6}}$ and $\underset{\sim}{\underline{b7}}$ arises from the following distinction:

$\underset{\sim}{\underline{b6}}$: Set of all b(j)'s who do not teach at least one of the members of the set a(i).

$\underset{\sim}{\underline{b7}}$: Set of all b(j)'s who do not belong to the set $\underset{\sim}{\underline{b1}}$ that teach at least one member of the set (ai), i.e. who do not teach any member of a(i).

It is very satisfying to know that such hair-splitting differences can be straightaway programed in MATLOG, once the logic is written in BVMF notation.

(b) Combination of two matrix relations

In the previous subsection 5(a), we had considered the combination of two matrices via the Boolean operations of sum and product as applied to relations. However, these are not the only ways in which two matrix relations can be combined. There are more possibilities which belong to the matrix type of operations rather than the Boolean type of operations. We shall describe these below with special reference to two problems and then consider the nature of such combinations more generally.

(i) Transitive combination of two matrix relations

Consider three sets A, B, C whose members are represented by the vectors $\underset{\sim}{a}(i)$, $\underset{\sim}{b}(j)$, $\underset{\sim}{c}(k)$, $i = 1$ to I, $j = 1$ to J, $k = 1$ to K, which are related by two successive relations as in (6):

$$\underset{\sim}{a}(i) \;\; \underset{\sim}{R1}\,(i,j) = \underset{\sim}{b}(j) \tag{6a}$$

$$\underset{\sim}{b}(j) \;\; \underset{\sim}{R2}\,(j,k) = \underset{\sim}{c}(k) \tag{6b}$$

Then the matrix relation directly connecting $\underline{a}(i)$ with $\underline{c}(k)$ is

$$\underset{\sim}{a}(i) \;\; \underset{\sim}{R3}(i,k) = \underset{\sim}{c}(k) \tag{7a}$$

where

$$|\underset{\sim}{R3}| = |\underset{\sim}{R1}| \; |\underset{\sim}{R2}| \tag{7b}$$

In fact, this type of combination of two matrices has already been defined in one of the MATLOG functions listed above, namely GMATPT whose description is as follows. The function giving $|\underset{\sim}{\underline{R3}}| = GMR3$, in terms of $|\underset{\sim}{\underline{R1}}| = GMR1$, $|\underset{\sim}{\underline{R2}}| = GMR2$ is

GMR3 = GMATPT(GMR1, GMR2, MI,MJ,MK)

Simple examples of GMATPT are:

$\underset{\sim}{\underline{R1}}$ = "parent of", $\underset{\sim}{\underline{R2}}$ = "parent of" $\mapsto$ $\underset{\sim}{\underline{R3}}$ = "grandparent of" (8a)

$\underset{\sim}{\underline{R1}}$ = "parent of", $\underset{\sim}{\underline{R2}}$ = "brother of" $\mapsto$ $\underset{\sim}{\underline{R3}}$ = "uncle of" (8b)

It is bovious that, having obtained the matrix $\underset{\sim}{\underline{R3}}(i,k)$, we can forget $\underset{\sim}{\underline{R1}}$ and $\underset{\sim}{\underline{R2}}$ from which this was derived, and consider it as a single matrix relation and manipulate it in all the processes described above. For example, $\underset{\sim}{\underline{R3}}^t$ will be the relation "nephew or niece of". It can be extended to any number of such successive relations connected in a transitive fashion. Therefore, this type of combination finds much application in predicate calculus.

<u>(ii) Tensor combination of two matrix relations</u>

This does not strictly come under the purview of this report, but is included here so as to make the list of possible ways in which two relations can be combined fairly complete. It is illustrated by the following examples. Consider three sets A, B1, B2 having the relations represented by the BVMF equations

$$\underset{\sim}{\underline{a}}\ \underset{\sim}{\underline{R1}} = \underset{\sim}{\underline{b1}}\ ,\quad \underset{\sim}{\underline{a}}\ \underset{\sim}{\underline{R2}} = \underset{\sim}{\underline{b2}} \tag{9a}$$

where

$$\underset{\sim}{\underline{a}} = \underset{\sim}{\underline{a}}(i),\ \underset{\sim}{\underline{b1}} = \underset{\sim}{\underline{b1}}(j),\ \underset{\sim}{\underline{b2}} = \underset{\sim}{\underline{b2}}(k) \tag{9b}$$

A good example is,

$$c.(i) = \text{child},\quad f(j) = \text{father},\quad m(k) = \text{mother} \tag{10a}$$

Then we can talk of the relation between a child and the "parents" (father and mother) of its parents, which we shall represent by $\underset{\sim}{\underline{\underline{p}}}(j,k)$. Then this relation takes the form

$$\underset{\sim}{\underline{c}}(i)\quad \underset{\sim}{\underline{R3}}(i,j,k) = \underset{\sim}{\underline{p}}(j,k) \tag{10b}$$

It is obvious that the entity $\underline{R}(i,j,k)$ is not a rectangular matrix, but a third order tensor in this particular case. More generally, a general tensor relation can be built up of elementary relations of the type given in (9), each of which requires a second order tensor, or rectangular matrix, for its definition. An introduction to such general tensor relations in Boolean algebra has been given in MR-59, and the implementation of such relations generally for practical problems will be considered later.

Two examples are given below for the relations "parent" standing for "either father or mother", and "parents" corresponding to "the pair consisting of father and mother" of child c(i). These are described in the Sections (c) and (d) below.

(c) Union of two matrix relations

The relation "parent of" is not a third order tensor relation, but the Boolean sum of two matrix relations. Taking (10a), if

$$c(i) \text{ has } i = 1 \text{ to } I \text{ , } f(j) \text{ has } j = 1 \text{ to } J, \; m(k) \text{ has } k = 1 \text{ to } K \tag{11a}$$

we can define $p(\ell)$ for "parent" with the domain

$$p(\ell) \text{ , } \ell = 1 \text{ to } L = J + K \tag{11b}$$

where

$$p(\ell) = f(j) \quad \text{for} \quad \ell = j \quad , \quad j = 1 \text{ to } J \tag{11c}$$

and

$$p(\ell) = m(k) \quad \text{for} \quad \ell = J + k \text{ , } \quad k = 1 \text{ to } K \tag{11d}$$

Then, in the extended domain (i, ℓ), the relations $\underset{\sim}{R1}'$ (father of) and $\underset{\sim}{R2}'$ (mother of), as in (12a,b)

$$\underset{\sim}{\underline{R1}}'(i,\ell) = |c(i)\rangle \otimes \langle p(\ell)|, \quad 1 \leqslant \ell \leqslant J \tag{12a}$$

$$\underset{\sim}{\underline{R2}}'(i,\ell) = |c(i)\rangle \otimes \langle p(\ell)|, \quad J + 1 \leqslant \ell \leqslant L \tag{12b}$$

will lead to $\underset{\sim}{\underline{R3}}'$ for "parent of", as in (13)

$$\underset{\sim}{\underline{R3}}'(i,\ell) = \underset{\sim}{\underline{R1}}'(i,\ell) \bigoplus \underset{\sim}{\underline{R2}}'(i,\ell) \tag{13}$$

This formalism can be suitably applied for all problems requiring the relation between the union of sets such as F and M into a single set $P = F \bigcup M$, and a third set such as C, in the above example.

(d) Expanded product of two matrix relations

On the other hand, the relation "parents of" is a tensor of the third order, obtained by the expansion of the two matrices $\underset{\sim}{\underline{R1}}(i, j)$ and $\underset{\sim}{\underline{R2}}(i, k)$ to yield $\underset{\sim}{\underline{R3}}(i,j,k)$ whose components R3(i,j,k) are given by (14).

$$R3(i,j,k) = R1(i, j) \times R2(i, k) \tag{14}$$

Then, Eq.(10b) holds for obtaining the pair p(j, k) = "f(j) and m(k)" of the child c(i). We shall consider the MATLOG statement for this in due course in a later more comprehensive report of tensor relations.

(e) Tensor relation $\underset{\sim}{R}(i,j,k)$ applied in three ways

It is to be noted, however, that formally, the tensor components R(i,j,k) correspond to a <u>mutual</u> relation between c(i), f(j), m(k), and we can obtain equally well the "mother-child" combinations who are related to a given father f(j) by the equation (15).

$$\underset{\sim}{f}(j)\ \underset{\sim}{R}(i,j,k) = \underset{\approx}{q}(i, k) \quad (\equiv \text{the combinations } c(i),\ m(k)) \qquad (15)$$

So also, the "father-child" combinations related to a set of given mothers $\underset{\sim}{m}(k)$ is given by Eq.(16),

$$\underset{\sim}{m}(k)\ \underset{\sim}{R}(i,j,k) = \underset{=}{r}(i, j) \qquad (16)$$

These also can be incorporated in programs, but will be deferred for a more comprehensive treatment of tensor relations.

6 . Generalized Clausal Relations

In all that has been considered so far, we have used only one type of relation, for an mxn matrix R(i, j), namely the standard form of a relation :

$$a_1 \wedge a_2 \wedge \ldots \wedge a_m \xrightarrow{R} b_1 \vee b_2 \vee \ldots \vee b_n \qquad (17)$$

in which R(i, j) is a general mxn matrix, including in particular, the case of the matrix corresponding to the clausal form, namely

$$(\underset{\sim}{\underline{R}} = \underset{\sim}{\underline{k}} \times \underset{\sim}{\underline{\ell}}) \equiv (GMZ = GDIRPT(GVK,GVL,MI,MJ)) \qquad (18)$$

As already mentioned, the MATLOG equation for (17) takes the forms (19a) and (19b) for unary and binary implementation of (17)

Unary : GUNPDT (GVA, GMZ, MI, MJ) = GVBP (19a)

Binary : GBTV(GVA, GMZ, GVB, MI, MJ) = BC (19b)

However, as considered in MR-57, (17) has three more analogues, leading to a set of four types of relations R1, R2, R3, R4 as below.

$$R1 : a_1 \wedge a_2 \wedge \ldots \wedge a_m \xrightarrow{R} b_1 \vee b_2 \vee \ldots \vee b_n \qquad (20a)$$

$$R2 : a_1 \vee a_2 \vee \ldots \vee a_m \xrightarrow{R} b_1 \vee b_2 \vee \ldots \vee b_n \qquad (20b)$$

$$R3 : a_1 \wedge a_2 \wedge \ldots \wedge a_m \xrightarrow{R} b_1 \wedge b_2 \wedge \ldots \wedge b_n \qquad (20c)$$

$$R4 : a_1 \vee a_2 \vee \ldots \vee a_m \xrightarrow{R} b_1 \wedge b_2 \wedge \ldots \wedge b_n \qquad (20d)$$

For implementing these in MATLOG, three additional subroutines are needed in GLOGIC, and one extra one, with four parts, in GCLOG. These are as follows:

Add under GLOGIC

(vii a) General conjunctive product (Type A) of two vectors

$$(a_1^c \otimes b_1 \oplus a_2^c \otimes b_2 \oplus \ldots \oplus a_m^c \otimes b_m)^c = c$$

GCJPTA(GVA,GVB, MI) = BC

BCMP(GSCPDT(GCOMP(GVA),GVB, MI)) = BC

(vii b) General conjunctive product (Type B) of two vectors

$$(a_1 \otimes b_1^c \oplus \ldots \oplus a_m \otimes b_m^c)^c = c$$

GCJPTB(GVA,GVB,MI) = BC

BCMP(GSCPDT(GVA,GCOMP(GVB), MI)) = BC

(viiia) Unary conjunctive product of a vector with a matrix

$$(\bigoplus_i (a_i^c \otimes z_{ij}))^c = b_j$$

GUNCJP(GVA,GMZ,MI,MJ) = GVB

GCOMP(GUNPDT(GCOMP(GVA), GVZ,MI,MJ)) = GVB

Add under GCLOG

(viii) Boolean truth value of a relation for given inputs

$\underset{\sim}{a}\ \underset{\sim}{Z}\ \underset{\sim}{b} = c$, for the four types of relations R1, R2, R3, R4. It is better to have four different functions, rather than using a variable having the values 1 to 4. Thus, we have

(viii a) Type R1

GBTVR1 (GVA,GMZ,GVB,MI,MJ) = BC

GUNPDT(GVA,GMZ,MI) = GVBP

GSCPDT(GVBP,GVB,MJ) = BC

(viii b) Type R2

GBTVR2(GVA,GMZ,GVB,MI,MJ) = BC

GUNCJP(GVA,GMZ,MI) = GVBPP

GSCPDT(GVBPP,GVB,MJ) = BC

(viii c) Type R3

GBTVR3(GVA,GMZ,GVB,MI,MJ) = BC

GUNPDT(GVA,GMZ,MI) = GVBP

GCJPTB(GVBP,GVB,MJ) = BC

(viii d) Type R4

GBTVR4(GVA,GMZ,GVB,MI,MJ) = BC

GUNCJP(GVA,GMZ,MI) = GVBPP

GCJPTB(GVBPP,GVB,MJ) = BC

The theory of these is given completely in MR-57, and Table 3 on page 27 therein contains the formulae which have been converted to MATLOG notation in this section.

7. Aristotle's Syllogisms in GLOGIC

As mentioned in MR-57, the generalized clausal relations of Types 1 and 2 can be expressed *via* quantifiers as follows:

Type R1 : $(\forall i)(ai) \xrightarrow{R} (\exists j)(bj)$ (I-type) (21a)

Type R2 : $(\exists i)(ai) \xrightarrow{R} (\exists j)(bj)$ (A-type) (21b)

where the notation A, I follows standard symbolism adopted for the names of syllogisms such as Barbara, Darii etc. (See MR 34).

Thus

Type A : All A are B ⟺ If there is an ai present, then (22a)
there is a bj present

Type I : Some A are B ⟺ If all ai are present, then (22b)
there is a bj present

To this may be added the two classical types, E and O, of categorical statements, as in (22c,d).

Type E : All A are not B ⟺ If there is an ai present, then
there is a non-bj present (22c)

Type O : Some A are not B ⟺ If all ai are present, then
there is a non-bj present (22d)

These correspond, in terms of quantifiers as :

Type R1 : $(\forall i)(ai) \xrightarrow{R} (\exists j)(\neg bj)$ (O-type) (21c)

Type R2 : $(\exists i)(ai) \xrightarrow{R} (\exists j)(\neg bj)$ (E-type) (21d)

An analysis of the logical content of (22a-d) indicates that the relation R in (21a-d) has the form ai = bj . However, without writing out in detail the matrix R, the statements (21a-d) can be taken over in QL-2 formulation, and the 18 categorical syllogisms that are valid, out of the 64 that can be stated, can be worked out via QL-2 algebra. (This will be discussed in detail in a later theoretical report.) We shall show how this can be deduced using MATLOG, as below.

From the form of the relations (21a-d), the four 3x3 matrices for categorical statements of the type A, E, I, O can be calculated (using (xvii) of MATLOG-2, MR-61) :

$$\text{Type } \underline{A} \;:\; QMZA = QMI,Q6,Q6 = QMO,Q5,Q6 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \tag{23a}$$

$$\text{Type } \underline{E} \;:\; QMZE = QMI,Q6,Q2 = QMO,Q5,Q2 = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{23b}$$

$$\text{Type } \underline{I} \;:\; QMZI = QMI,Q1,Q6 = QMO,Q2,Q6 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{23c}$$

$$\text{Type } \underline{O} \;:\; QMZO = QMI,Q1,Q2 = QMO,Q2,Q2 = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{23d}$$

We shall first show that the two classical syllogisms which are basic to the working out of all valid ones in Aristotle's list, namely <u>Barbara</u> and <u>Darii</u>, follow from the form of the matrices in (23a-d). In words, these two are as follows:

$$\text{All A is B, All B is C} \Longleftrightarrow \text{All A is C} \quad (\underline{\text{Barbara}}) \qquad (24a)$$

$$\text{Some A is B, All B is C} \Longleftrightarrow \text{Some A is C} \quad (\underline{\text{Darii}}) \qquad (24b)$$

The corresponding MATLOG equations to be checked are :

QMATPT(QMZA, QMZA) = QMZA (25a)

QMATPT(QMZI, QMZA) = QMZI (25b)

These are seen to be valid from (23a,c).

Other syllogisms follow similarly. For example, <u>Celarent</u> follows as in (25c), and <u>Ferio</u> from (25d).

QMATPT(QMZA, QMZE) = QMZE (25c)

QMATPT(QMZI, QMZE) = QMZO (25d)

So also, the non-existence of any syllogisms of the Aristotelean type other than the 18 listed in MR-34 can be proved by showing that the six others in Table 3, page 33 of MR-34 (other than Sl. No.1, <u>Barbara</u> and Sl. No. 3, <u>Darii</u>) lead to no definite conclusions. Thus, as will be seen from (26a-f) below, the output matrix is QMDDD in all these cases, and it is a matrix that will give an output Q8 = (1 1 1), corresponding to the state △ for universal doubt, whatever may be the input (except the impossible state (0 0 0)).

Sl. No. 2: QMATPT(QMZE, QMZA) = QMDDD (26a)

Sl. No. 4: QMATPT(QMZO, QMZA) = QMDDD (26b)

Sl. No. 5: QMATPT(QMZA, QMZI) = QMDDD (26c)

Sl. No. 6: QMATPT(QMZE, QMZI) = QMDDD (26d)

Sl. No. 7: QMATPT(QMZI, QMZI) = QMDDD (26e)

Sl. No. 8: QMATPT(QMZO, QMZI) = QMDDD (26f)

where

$$\text{QMDDD} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \qquad (26g)$$

These simple calculations using MATLOG give a proof of the consistency and completeness of all the syllogisms of the Aristotlean type in classical logic. The precise conditions in BVMF, under which the additional 18 syllogisms of the "new" type listed in MR-34 are valid, have not yet been worked out.

8. Mixed statements containing quantified and unquantified terms via GLOGIC

The formulae given below are based on the treatment in MR-53. In quantifier algebra (QL-2) we can have statements of the type QSN and SNQ with 3x2 matrices and 2x3 matrices respectively. These are best treated using the GLOGIC formulae developed in this report. We only give below the MATLOG statements which will serve for QSN unary and binary products and SNQ unary and binary products, with the definition of QSN and SNQ matrices *via* the logical relations represented by them. They are not given any number since, in each case, a MATLOG statement in GLOGIC using the formulae given in Sections 2 and 3 will serve the purpose. They are listed below.

<u>QSN – unary product</u> : $\underset{\sim}{a}\,\underset{\sim}{\underline{Z}} = \underline{\underline{b}}$

GUNPDT(QVA,QSMZ, 3,2) = SVB

<u>QSN – binary product</u> : $\underset{\sim}{a}\,\underset{\sim}{\underline{Z}}\,\underline{\underline{b}} = c$

GBTV(QVA,QSMZ,SVB, 3,2) = BC

<u>QSN – relational matrices</u> : $\underset{\sim}{k} \times \underline{\underline{\ell}}$, $\underset{\sim}{k} + \underline{\underline{\ell}}$, $\underset{\sim}{k} \equiv \underline{\underline{\ell}}$, $\underset{\sim}{k} \Longrightarrow \underline{\underline{\ell}}$

QSMZ = QSMX, QVK,SVL = GMX,GVK,GVL

where GVK = QVK, GVL =SVL

<u>SNQ – unary product</u> : $\underline{\underline{a}}\,\underset{\sim}{\underline{Z}} = \underset{\sim}{b}$

GUNPDT(SVA,SQMZ, 2, 3) = QVB

<u>SNQ – binary product</u> : $\underline{\underline{a}}\,\underset{\sim}{\underline{Z}}\,\underset{\sim}{b} = c$

GBTV(SVA,SQMZ,QVB, 2,3) = BC

<u>SNQ – relational matrices</u> : $\underline{\underline{k}} \times \underset{\sim}{\ell}$, $\underline{\underline{k}} + \underset{\sim}{\ell}$, $\underline{\underline{k}} \equiv \underset{\sim}{\ell}$, $\underline{\underline{k}} \longrightarrow \underset{\sim}{\ell}$

SQMZ = SQMX,SVK,QVL = GMX,GVK,GVL,

where GVK = SVK, GVL = QVL

Other logical functions such as relative truth value, SNS truth value of a relation etc., can also be formulated using suitable subroutines from GLOGIC. Details are not given.

As discussed in MR-53, it can happen that the input for a quantified term in a relation is a non-quantified term, and vice-versa, and similarly for the output. However, the necessary formulae for this purpose which were developed in MR-53 are based on <u>logic</u> and are expressible in Boolean <u>algebra</u> according to the logical conditions that are imposed. Since this is a problem in logic, it is deferred for consideration in the detailed report on the application of BVMF for practical problems.

9. Comment on the combination of two relations in general

In Part II of MATLOG series (MR-61), it was mentioned in Section 2, page 12, that, in general, the conjunction of the Boolean truth values of two relations is not necessarily equivalent to the Boolean truth value of the conjunction of the two relations, while the corresponding result is always true for disjunctions, and that this will be considered in Part III dealing with GLOGIC. This has in fact been done in Section 5, page 20 of this report, where, in a practical example, the non-equivalence of the two types of conjunction, as also the equivalence of the two types of disjunction, has been illustrated. A general proof of this is reserved for a theoretical report dealing with theory of relations in general. However, the proof that the two ways of combining two relations will lead to the same result for the output when one of the conditions (a), (b), (c) of Section 4, page 35, of MATLOG-1 (MR-60) is satisfied, can be given, as mentioned therein, for GLOGIC, and hence for QLOGIC and SNSLOG. This is briefly indicated below.

(a) Unary relations and Boolean truth values of binary relations

Following the notation employed in this part, we shall consider, in particular, the following unary relations,

$$\underset{\sim}{\underline{a}}\;\underset{\sim}{\underline{\underline{R1}}} = \underset{\sim}{\underline{b1}}\;, \qquad \underset{\sim}{\underline{a}}\;\underset{\sim}{\underline{\underline{R2}}} = \underset{\sim}{\underline{b2}} \qquad (27a,b)$$

which lead,by conjunction and disjunction of the relations R1 and R2, to (28a,b):

$$\underset{\sim}{\underline{a}}\;(\underset{\sim}{\underline{\underline{R1}}} \otimes \underset{\sim}{\underline{\underline{R2}}}) = \underset{\sim}{\underline{c}}\;, \quad \underset{\sim}{\underline{a}}\;(\underset{\sim}{\underline{\underline{R1}}} \oplus \underset{\sim}{\underline{\underline{R2}}}) = \underset{\sim}{\underline{d}} \qquad (28a,b)$$

So also, by conjunction and disjunction of the output vectors $\underset{\sim}{\underline{b1}}$ and $\underset{\sim}{\underline{b2}}$ of (27a,b) we obtain the outputs of (29a,b):

$$\underset{\sim}{\underline{b1}} \otimes \underset{\sim}{\underline{b2}} = \underset{\sim}{\underline{c}}'\;, \quad \underset{\sim}{\underline{b1}} \oplus \underset{\sim}{\underline{b2}} = \underset{\sim}{\underline{d}}' \qquad (29a,b)$$

We shall prove below that

$$\underset{\sim}{\underline{c}} \neq \underset{\sim}{\underline{c}}' \text{ in general, while } \underset{\sim}{\underline{d}} = \underset{\sim}{\underline{d}}' \text{ always} \qquad (30a,b)$$

The proof is from pure Boolean algebra and does not involve any logical considerations. It follows from the definition of the unary matrix product contained in Eqs.(27a,b). Thus, with the notation adopted in this report, for a given input a(i), we have*

$$a(i) \otimes R1(i,j) = b1(j), \quad a(i) \otimes R2(i,j) = b2(j), \quad j = 1 \text{ to } n \qquad (31a,b)$$

* In the succeeding equations of this section, the symbol $\otimes$ is omitted wherever it is obvious.

Then, it necessarily follows that

$$a(i) \ (R1(i,j) \otimes R2(i,j)) = b1(j) \otimes b2(j) \qquad (32)$$

and similarly for the Boolean sum. However, if the vector $\underset{\approx}{a}$ is not a basic vector, with $a(i) = 1$ for a single i and equal to 0 for all other $i = 1$ to m, but is a general vector with more than one component, $a(i)$, $= 1$, then Eq.(32) does not follow, and it takes the form of non-equality of (33) :

$$\bigoplus_{i=1}^{m} a(i) \ (R1(i,j) \otimes R2(i,j)) \not\Leftrightarrow b1(j) \otimes b2(j), \quad j = 1 \text{ to } n \qquad (33)$$

On the other hand, because of the associative property of the Boolean sum, we obtain (34) in support of Eq.(30b), which can be written in the form (35):

$$\bigoplus_{i=1}^{m} a(i) \ (R1(i,j) \oplus R2(i,j)) \Longleftrightarrow b1(j) \oplus b2(j), \quad j = 1 \text{ to } n \qquad (34)$$

$$\underset{\approx}{a} \ (\underset{\approx}{\underline{R1}} \bigoplus \underset{\approx}{\underline{R2}}) = \underset{\approx}{a} \ \underset{\approx}{\underline{R1}} \bigoplus \underset{\approx}{a} \ \underset{\approx}{\underline{R2}} \qquad (35)$$

Thus, we have obtained a general proof of the result, associated with the particular examples given in Section 5, showing the non-equivalence for conjunction, and equivalence for disjunction, of the two ways of performing this logical operation with two relations applied to the same input. For the same reasons as given above, it can be shown that (36a,b) are valid for the Boolean truth value of two binary relations, applied to the same input and output in two different ways, as indicated there.

$$(\utilde{\underline{a}}\ \ \utilde{\underline{R1}}\ \ \utilde{\underline{b}}) \bigvee (\utilde{\underline{a}}\ \ \utilde{\underline{R2}}\ \ \utilde{\underline{b}}) \iff \utilde{\underline{a}}\ (\utilde{\underline{R1}} \oplus \utilde{\underline{R2}})\ \utilde{\underline{b}} \qquad (36a)$$

$$(\utilde{\underline{a}}\ \ \utilde{\underline{R1}}\ \ \utilde{\underline{b}}) \bigwedge (\utilde{\underline{a}}\ \ \utilde{\underline{R2}}\ \ \utilde{\underline{b}}) \not\Leftrightarrow \utilde{\underline{a1}}\ (\utilde{\underline{R1}} \otimes \utilde{\underline{R2}})\ \utilde{\underline{b}} \qquad (36b)$$

This follows for the reason that $\bigvee$ and $\bigwedge$ in these equations are represented by the Boolean operations $\oplus$ and $\otimes$, so that (36a) follows from the associative property of the Boolean sum, while no such result follows when Boolean sum and Boolean product occur together in an expression as in (36b). As mentioned in Section 5, the MATLOG equations for these are quite different and are not expected to be equal, but the equivalence holds in the particular case of disjunction for the reason mentioned above. The extension to SNS truth values is discussed below.

<u>(b) Conditions for conjunction identity</u>

Before considering this, we shall first consider statements (a),(b),(c) of MR-60 mentioned above, under which the identity of the two ways of performing the conjunction of two relations is obtained. With slight change in nomenclature, we have to show that $\underset{\approx}{c} = \underset{\approx}{c}'$ under the following three conditions.

$$\left.\begin{array}{ll} \text{(a)} & |\underset{\sim}{R1}| \supseteq |\underset{\sim}{R2}|, \text{ or } |\underset{\sim}{R1}| \subseteq |\underset{\sim}{R2}| \\ \text{(b)} & |\underset{\sim}{R1}| = |\underset{\sim}{R2}^{c}|, \text{ which is equivalent to } |\underset{\sim}{R1}^{c}| = |\underset{\sim}{R2}| \\ \text{(c)} & |\underset{\sim}{R1}| \otimes |\underset{\sim}{R2}| = |\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})| \end{array}\right\} \quad (3$$

We shall prove the first and the third of these, while the second condition (b) is found to be inaccurate. (This arose because of a mistake in writing that $\underline{\underline{P}}^{c} = \underline{\underline{Q}}$ in page 32 of MR-60)

Condition (a)

Taking the first condition, it follows that $R1(i,j) \Longrightarrow R2(i,j)$, which means that the set of 1's in $R1(i,j)$ contains all the elements having the value 1 in the set $R2(i,j)$. Under these conditions, the l.h.s of (33) becomes

$$\bigoplus_{i=1}^{m} a(i) \ R2(i,j) = b2(j)$$

Under the same conditions, we also obtain the result that $b1(j) \Longrightarrow b2(j)$, so that the r.h.s of (33) is also $b2(j)$ as in (37).

Similarly, for the second condition, it follows that $R2(i,j) \Longrightarrow R1(i,j)$ and $b2(j) \Longrightarrow b1(j)$, so that both the l.h.s and the r.h.s of (33) become equal to $b1(j)$, and hence equal to one another.

Condition (b)

On applying this condition to Eq.(32), we obtain the result that the r.n.s is a null vector and that tnis is not modified in any way as a consequence of the multiple sum in the l.h.s of (33). On the other hand, in the r.h.s of (33), $b1(j)$ is noτ, in general, equal to $b2^{c}(j)$ when the multiple sum is applied twice over, as in (38). Thus,

$$c'(j) = b1(j) \otimes b2(j) = \Big(\bigoplus_{i=1}^{m} a(i)\ R1(i,j)\Big) \otimes \Big(\bigoplus_{i=1}^{m} a(i)\ R2(i,j)\Big) \tag{38}$$

is not necessarily equal to 0 for all j, as is the case for $c(j)$. It can in fact be verified that, even for SNS,

$$|R1| = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \quad , \quad |R2| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{39}$$

do not lead to an identity for (33). The condition (b) is therefore insufficient as a general rule for (33) to be an equality. A complete output table of Problem 4A of MATLOG-1 has been prepared* which shows that condition (a) is valid in a

*See Report MR—65 for details.

vast majority of cases and only six examples are not derivable by this condition. Of these, only two possibilities, as in (39a,b)

$$|R1| = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} , \quad |R2| = \begin{pmatrix} 1 & 1 \\ 0 & \end{pmatrix} , \quad |R| = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad (39a)$$

$$|R1| = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} , \quad |R2| = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} , \quad |R| = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad (39b)$$

satisfy the condition (c). For the remaining four cases, both $|R1|$ and $|R2|$ have two 1's in a row and two 1's in a column, or <u>vice versa</u>. These have logical properties of the type "$\underline{\underline{a}}$ is always T(or F)" and "$\underline{\underline{b}}$ is always T(or F)", so thatt from pure logic, if both of them are simultaneously true, we get the result "$\underline{\underline{a}}$ = T and $\underline{\underline{b}}$ = T", "$\underline{\underline{a}}$ = T and $\underline{\underline{b}}$ = F", etc., whose matrices are of the form $A(k, \ell)$, $k, \ell = 1, 2$. So we replace the condition (b) in (37) by the following:

(b) Both $|R1|$ and $|R2|$ are singular matrices, one with two 1's in a row and the other with two 1's in a column.

With these three conditions (a), (b), (c), of (37), all the cases where the conjunction identity is satisfied are covered, and they can also be generalized to QLOGIC and GLOGIC as described in the next section 9(c).

Condition (c)

This is best proved for SNS first, and generalized to GLOGIC. We consider first the two cases $|E(1,\ 1)|$ and $|E(1,\ 2)|$ as given in (40):

$$|E(1,\ 1)| = |E| = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \ , \ |E(1,\ 2)| = |N| = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad (40)$$

For each of these, there is only one combination of $|R1|$ and $|R2|$ which will lead to their Boolean product being equal to $|E|$ or $|N|$ as the case may be. We shall prove each of these cases. The formulae to be proved are:

$$(a_1 \quad a_2) \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \overset{\rightarrow}{\bigotimes} (a_1 \quad a_2) \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = (a_1 \quad a_2) \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \text{ for } \underline{\underline{E}} \qquad (41a)$$

and

$$(a_1 \quad a_2) \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \bigotimes (a_1 \quad a_2) \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} = (a_1 \quad a_2) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \text{ for } \underline{\underline{N}} \qquad (41b)$$

It is readily verified that (41a) and (41b) are satisfied as shown in (42a,b):

$$(a_1 \quad a_1 \oplus a_2) \bigotimes (a_1 \oplus a_2 \quad a_2) = (a_1 \quad a_2) \qquad (42a)$$

$$(a_2 \quad a_1 \oplus a_2) \bigotimes (a_1 \oplus a_2 \quad a_1) = (a_2 \quad a_1) \qquad (42b)$$

Similarly, we can work out in detail that the equality holds for all combinations (k, ℓ) of SNS. A table of these is, in fact, given in MR-60, page 39, dealing with Problem 4B.

(c) Extension to GLOGIC

The condition (a) in (37), for the equality of the two ways of performing conjunction, is followed quite generally in GLOGIC, and hence also in SNS and QL-2, to which it reduces when we put $m = n = 2$ and $m = n = 3$, respectively. Thus $\underset{\approx}{c} = \underset{\approx}{c}'$ is valid under condition (a), for a general mxn Boolean matrix.

On the other hand, condition (c) has been proved in (42) only for SNS algebra employing 2x2 matrices. We shall now indicate how this can be generalized to GLOGIC when only logical connectives of the type $\underset{\approx}{a}\ \underset{\approx}{Z}(\underset{\approx}{k},\ \underset{\approx}{\ell})\ \underset{\approx}{b}$ are considered (where $Z = A, O, I$ or E). This is possible since, for logical relations of the type $\underset{\approx}{k} \wedge \underset{\approx}{\ell}$, $\underset{\approx}{k} \vee \underset{\approx}{\ell}$, $\underset{\approx}{k} \Longrightarrow \underset{\approx}{\ell}$, $\underset{\approx}{k} \Longleftrightarrow \underset{\approx}{\ell}$, the relevant mxn matrix of the relation takes the form given below in (43a to d), by a suitable rearrangement of the indices $i = 1$ to m, and $j = 1$ to n.

$$\wedge \longleftrightarrow \begin{array}{cc} \begin{array}{cc} \longleftarrow \ell \longrightarrow & \longleftarrow n-\ell \longrightarrow \end{array} & \\ \left(\begin{array}{c|c} 1\text{'s} & 0\text{'s} \\ \hline 0\text{'s} & 0\text{'s} \end{array}\right) & \begin{array}{c} \updownarrow k \\ \updownarrow m-k \end{array} \end{array} , \quad \vee \longleftrightarrow \left(\begin{array}{c|c} 1\text{'s} & 1\text{'s} \\ \hline 1\text{'s} & 0\text{'s} \end{array}\right) \qquad \text{(43a,b)}$$

$$\Longrightarrow \ \longleftrightarrow \left(\begin{array}{c|c} 1\text{'s} & 0\text{'s} \\ \hline 1\text{'s} & 1\text{'s} \end{array}\right) , \quad \Longleftrightarrow \ \longleftrightarrow \left(\begin{array}{c|c} 1\text{'s} & 0\text{'s} \\ \hline 0\text{'s} & 1\text{'s} \end{array}\right) \qquad \text{(43c,d)}$$

These are seen to have a close correspondence with the structure of the 2x2 matrices in SNS, for the four connectives A, O, I, E respectively.

From Boolean matrix algebra, it is obvious that the four relations represented by the matrices in (43a,b,c,d) will have properties identical with the four SNS matrices for A, O, I, E, with the only difference that they refer to $\underset{\approx}{a}\ \underset{\approx}{Z}\ \underset{\approx}{b}$, as distinct from $\underline{\underline{a}}\ \underline{\underline{Z}}\ \underline{\underline{b}}$ for SNS. In the case of 2x2 matrices, if a_i and b_j are related by $Z_{ij} = 1$, then the unary relation $\underline{\underline{a}}\ \underline{\underline{Z}} = \underline{\underline{b}}$ leads to b_j if a_i is input. This same sentence can be carried over to the general relation in GLOGIC. However, in SNS, there can only be one a_i and one b_j in the four quadrants marked in (43a,b,c,d) while there can be a larger number of them (such as $k\ell$, $k(n - \ell)$ etc.) in GLOGIC. A little reflection will show that we have in this manner completely transferred the <u>logical</u> properties of the relations $\underset{\approx}{k} \wedge \underset{\approx}{\ell}$,..., $\underset{\approx}{k} \Longleftrightarrow \underset{\approx}{\ell}$, connecting the two sets represented by $\underset{\approx}{k}$ and $\underset{\approx}{\ell}$, into the matrix formalism,by using this procedure.

Therefore, if we consider the condition (c) mentioned in (37) and worked out in (43) for SNS, the whole proof from (40) to (42) can be carried over into GLOGIC, with $\underline{\underline{E}}(k, \ell)$ being replaced by $\underset{\approx}{E}(\underset{\approx}{k}, \underset{\approx}{\ell})$, subject to the generalizations mentioned above. Thus, we can prove, for instance, that (44) is valid for GLOGIC.

$$(\underset{\approx}{a} \; \underset{\approx}{I} \; \underset{\approx}{b}) \wedge (\underset{\approx}{a}^c \; \underset{\approx}{I} \; \underset{\approx}{b}^c) = \underset{\approx}{a} \; \underset{\approx}{E} \; \underset{\approx}{b} = (\underset{\approx}{a} \; \underset{\approx}{A} \; \underset{\approx}{b}) \vee (\underset{\approx}{a}^c \; \underset{\approx}{A} \; \underset{\approx}{b}^c) \qquad (44)$$

This is also applicable to logical connectives of the type involving a clausal relation, and their generalizations considered in earlier sections, for which most of the results worked out in SNS algebra can be taken over. On the other hand, as already mentioned, the results proved in (31) to (36) and for condition (a) of (37) are valid, not only for logical matrix operators, but also for any general m x n matrix occurring for the relation in GLOGIC.

(d) SNS truth values

The extension of the above considerations to SNS truth values is fairly straightforward. For this purpose, we have only to show the identity of $\underline{\underline{c}}$ of (45a) and $\underline{\underline{c}}'$ of (45b) for the conjunction of two relations $\underset{\approx}{R1}$ and $\underset{\approx}{R2}$.

$$\underline{\underline{c}} = \underline{\underline{t}}(\underset{\approx}{a} \; \underset{\approx}{R1} \; \underset{\approx}{b}) \wedge \underline{\underline{t}}(\underset{\approx}{a} \; \underset{\approx}{R2} \; \underset{\approx}{b}) \qquad (45a)$$

$$\underline{\underline{c}}' = \underline{\underline{t}}(\underset{\approx}{a} \; (\underset{\approx}{R1} \otimes \underset{\approx}{R2}) \; \underset{\approx}{b}) \qquad (45b)$$

It can be readily shown that this equality (using GSTV) is equivalent to two equations (46a) and (46b) given below.

$$\langle \underset{\sim}{\underline{a}} | \underset{\sim}{\underline{\underline{R1}}} | \underset{\sim}{\underline{b}} \rangle \otimes \langle \underset{\sim}{\underline{a}} | \underset{\sim}{\underline{\underline{R2}}} | \underset{\sim}{\underline{b}} \rangle = \langle \underset{\sim}{\underline{a}} | \underset{\sim}{\underline{\underline{R1}}} \circledcirc\!\!\!\!\otimes \underset{\sim}{\underline{\underline{R2}}} | \underset{\sim}{\underline{b}} \rangle \qquad (46a)$$

$$\langle \underset{\sim}{\underline{a}} | \underset{\sim}{\underline{\underline{R1^c}}} | \underset{\sim}{\underline{b}} \rangle \oplus \langle \underset{\sim}{\underline{a}} | \underset{\sim}{\underline{\underline{R2^c}}} | \underset{\sim}{\underline{b}} \rangle = \langle \underset{\sim}{\underline{a}} | \underset{\sim}{\underline{\underline{R1^c}}} \circledcirc\!\!\!\!\oplus \underset{\sim}{\underline{\underline{R2^c}}} | \underset{\sim}{\underline{b}} \rangle \qquad (46b)$$

Of these, the second equation (46b) for disjunction is universally true, following from the result (35). Therefore, the condition for the equality of the SNS truth values of (45a) and (45b) is the same as that for the equality to hold between the l.h.s and r.h.s for the Boolean truth values (GBTV) in (46a). This has been discussed above, and we have listed the three conditions (a), (b), (c) in Section 9(b) above. Therefore, if one of these conditions is satisfied, not only is there a conjunction identity for Boolean truth values, but also for SNS truth values defined in (45a,b).

Since we have proved this for a general relation denoted by m x n matrices, it is equally true for QL-2 and SNS logic. Therefore, all the results obtained in the problems discussed in MR-60 and 61 become explainable. The most interesting consequence from the point of view of the logic, is that the conjunction of two relations is not a uniquely definable property, and that its logical consequences will depend upon whether the conjunction is applied to the two relations as such, or to the truth values of the consequences of each of the two relations applied individually. This will be pursued further in a theoretical paper. (See MR-65)

10. Comparison of GSTV and GSTV2

This section is what has been referred to as Appendix to MR-62 in the previous two parts MR-60 and 61, with special reference to the analysis of Problems 4B, 4C of the former and Problems 8B, 8C of the latter. The discussion has also relevance to Problems 1A and 4A of the former and Problem 7 of the latter. They form a natural extension of the previous section 9 of this report, with special reference to the differences between the application of GSTV and GSTV2 to calculate truth values. We shall first discuss the essential feature with reference to SNS logic and then indicate how these can be extended to QLOGIC and to GLOGIC in general.

(a) Difference between SBTV and SBTV2

Although the two functions SBTV and SBTV2 have not been specially defined in MR-60 dealing with SNS logic, they are particular cases of GLOGIC where GSTV and GSTV2 have been defined. The crucial differences between the two, for relations expressible by logical connectives in propositional calculus, can be traced to the following example of Boolean truth values, calculated _via_ the matrix formalism, and _via_ the relative truth values, for the simple case of a disjunction $\underline{\underline{a}} \vee \underline{\underline{b}}$. For the case when $\underline{\underline{b}} = (0\ 0) = X$, the following two results (47a,b) are obvious.

$$(1\quad 0)\begin{pmatrix}1 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}0\\ 0\end{pmatrix} = 0 \quad (\text{SBTV}) \qquad (47a)$$

$$(1\quad 0)\begin{pmatrix}1\\ 0\end{pmatrix} \quad (1\quad 0)\begin{pmatrix}0\\ 0\end{pmatrix} = 1 \quad 0 = 1 \quad (\text{SBTV2}) \qquad (47b)$$

Thus, for the relation $\underline{\underline{a}} \vee \underline{\underline{b}}$, and the inputs $\underline{\underline{a}} = T$ and $\underline{\underline{b}} = X$, the truth value calculated using the MATLOG formulation of BVMF is 0, while that calculated using relative truth values as in the "FORTRAN Program NYAYA2" (MR-10) is 1. Even at the time that NYAYA was developed, we had noticed that the formulae led to a non-X state for the SNS truth value of $\underline{\underline{a}} \vee \underline{\underline{b}}$ for the inputs $\underline{\underline{a}} = T$, $\underline{\underline{b}} = X$, even though one of the inputs is the impossible state which would intuitively indicate that the relation should also have the contradictory state X. This was taken care of by introducing an additional condition in all NYAYA formulae, (except union), namely "X-priority check", which made the truth value X if either $\underline{\underline{a}}$ or $\underline{\underline{b}} = X$ (see MR-12, p 40 and MR-16, p9, in particular). With this condition, it can be readily verified that SBTV and SBTV2 will agree for all inputs and outputs and for all matrix relations in SNS logic.

In fact, when MATLOG was first developed and the first program for MATLOG was written in MR-18, we noticed that the X-priority check was no longer necessary for truth values, and therefore, this has never been mentioned explicitly thereafter, until in MR-62 PLOGIC formulae were discussed, where, once again, this was found to be necessary. For instance, we have put the condition

$$\text{IF} \quad \text{QVA . OR . QVB} = \text{Q8} \;, \quad \text{QVC} = \text{Q8} \qquad (48)$$

Therefore it suggests itself that, for truth values requiring the condition that impossible as input always leads to impossible as output (except for union), BVMF with matrix multiplication can always be employed, including the case of union in SUNION, GUNION and QUNION.

(b) Extension to SSTV2, QSTV2, GSTV2

The question arises as to why SBTV2, and the derived truth value formulae SSTV2, QSTV2 and GSTV2, are necessary. This is best made clear by taking the general example of GBTV2 corresponding to the disjunction $\underset{\sim}{\underline{O}}(\underset{\sim}{\underline{k}}, \underset{\sim}{\ell}) = \underset{\sim}{\underline{k}} \cdot \vee \underset{\sim}{\ell}$, which means, in multivalued logic corresponding to sets $\mathcal{A}$ and $\mathcal{B}$ having the subsets A_K and B_ℓ, that "A_K or B_ℓ is true" .

In this case, if B_ℓ is a null set , since A_K is not a null set, if one element of this is present, then the given relation is true. This feature can be obtained only by using GBTV2, and not by GBTV, which will automatically make the output false for all inputs if $B = \phi = (0 \;\; 0 \ldots 0 \;\; 0)$. Therefore, the converse of the X-priority check is demanded for set theory and multivalued logic. The matrix formulation is universally applicable so long as neither A nor B is a null set. If one of them is a null set <u>and</u> the relation is a logical disjunction of "A_K or B_ℓ" , then GBTV2 and the analog GSTV2 must be employed. In fact, even for the sets A and B having only two members, or three members, and the vector $\underline{\underline{a}}$, or $\underset{\sim}{a}$, is not a truth value, but represents a general 2-vector or 3-vector $(\underset{\sim}{a})$ for sets, the same considerations hold.

These are not defects of BVMF, but rather very much to its credit, as it is possible to distinguish clearly the application of m x n matrices for all $m, n \geqslant 2$, to the truth values in SNS and QL, and to set theory in general, for all applications. It might even happen, in set theory, that the occurrence of the null set is an impossibility, in which case, this can be incorporated in a suitable manner in the formulae. In general, if the conditions of the problem are precisely stated, it is always possible to construct a vector-matrix formula to take care of the logical features of the problem, and then we can make it computer implementable for solution.

If these features are understood, then the solutions to all the problems mentioned above become understandable. Some of these are discussed below, and in MR-65, and full tables of data are contained there.

(c) Explanation of the differences between SSTV and SSTV2 for $\underline{\underline{E}}(k,\ell)$

Analogous to (47a,b) above, the two BVMF formulae for $\underline{\underline{a}} \wedge \underline{\underline{b}}$ are as follows.

$$(1 \quad 0)\begin{pmatrix}1 & 0\\0 & 0\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix} = 0 \quad \text{(SBTV)} \qquad (49a)$$

$$(1 \quad 0)\begin{pmatrix}1\\0\end{pmatrix} \otimes (1 \quad 0)\begin{pmatrix}0\\0\end{pmatrix} = 1 \otimes 0 = 0 \quad \text{(SBTV2)} \qquad (49b)$$

In this case, the two agree. In fact, as a general proposition, it can be shown that all conjunctions lead to agreement between SBTV and SBTV2, while for disjunctions if either $\underline{\underline{a}}$ or $\underline{\underline{b}}$ is X, then it is possible that SBTV and SBTV2 disagree.

Combining these two results in (47) and (49), we can make the following general statement of the likely differences between SSTV and SSTV2.

"For all conjunctions $\underset{=}{a} \wedge \underset{=}{b}$, if either $\underset{=}{a}$ or $\underset{=}{b}$ is X, then it is possible that SSTV2 gives F while SSTV gives X" . (50a)

"For all disjunctions $\underset{=}{a} \vee \underset{=}{b}$, if either $\underset{=}{a}$ or $\underset{=}{b}$ is X, it is possible that SSTV2 leads to T, while SSTV always leads to X". (50b)

This covers all of Problems 1A and 4A of Part I and Problem 7 of Part II.

In fact, the same considerations, extended to GLOGIC (as indicated in theory in MR-65) also explain the conditions under which SSTV and SSTV2 formulae give different results, for $\underset{=}{a}\,\underset{=}{E}\,\underset{=}{b}$ in SNS, and also correspondingly for $\underset{\sim}{a}\,\underset{\sim}{E}\,\underset{\sim}{b}$ in GL-2, and $\underset{\approx}{a}\,\underset{\approx}{E}\,\underset{\approx}{b}$ in GLOGIC. We shall not describe this in detail, but give some illustrative tables which, when proved, will make it clear. Since tables illustrating Problems 1A and 4A of Part I, and Problem 7 of Part II, are given in those parts, and as the complete output for Problems 4C and 8C will be given in MR-65, we shall only give selected tables from the latter two for illustration below. Thus, Table 5(a) contains the computer outputs for $\underline{\underline{E}}(1, 1)$ and $\underline{\underline{E}}(1, 2)$ for SSTV2 implementation of the mathematical formulae (51a,b), for $\underline{\underline{E}}(1, 1)$, and the corresponding formula generated for $\underline{\underline{E}}(1, 2)$ by replacing $\underline{\underline{b}}$ by $\neg\underline{\underline{b}}$.

$$\underline{\underline{a}}\ \underline{\underline{E}}\ \underline{\underline{b}} = (\underline{\underline{a}}\ \underline{\underline{A}}\ \underline{\underline{b}}) \bigvee (\neg\underline{\underline{a}}\ \underline{\underline{A}}\ \neg\underline{\underline{b}}) \qquad (51a)$$

$$\underline{\underline{a}}\ \underline{\underline{E}}\ \underline{\underline{b}} = (\underline{\underline{a}}\ \underline{\underline{I}}\ \underline{\underline{b}}) \bigwedge (\neg\underline{\underline{a}}\ \underline{\underline{I}}\ \neg\underline{\underline{b}}) \qquad (51b)$$

The first two columns correspond to the inputs $\underline{\underline{a}}$, $\underline{\underline{b}}$. The third column is a common output of (51a) and (51b) obtained by SSTV, while the entries in the 4th and 5th column are respectively those obtained with (51a) and (51b) using SSTV2 implementation, which differ from SSTV. It will be noticed that such differences occur only when either $\underline{\underline{a}}$ or $\underline{\underline{b}}$ is X and also that the differring entry is always F for the fourth column corresponding to (51a) and T for the fifth column corresponding to (51b). This feature is true for all $\underline{\underline{E}}(k, \ell)$, as will be seen from the other examples given in Tables 5(b,c). A complete output for this problem is included in MR-65.

Table 5. Check of Problem 4C for E(1, 1) and E(1, 2)

(a) Both k and ℓ not 3, or 4

```
     PROB  4C    (K,L)  =  (1,1)
 T   T           T
 T   F           F
 T   D           D
 T   X           X
 F   T           F
 F   F           T
 F   D           D
 F   X           X
 D   T           D
 D   F           D
 D   D           D
 D   X           X    F     T
 X   T           X
 X   F           X
 X   D           X    F     T
 X   X           X
```

```
     PROB  4C    (K,L)  =  (1,2)
 T   T           F
 T   F           T
 T   D           D
 T   X           X
 F   T           T
 F   F           F
 F   D           D
 F   X           X
 D   T           D
 D   F           D
 D   D           D
 D   X           X    F     T
 X   T           X
 X   F           X
 X   D           X    F     T
 X   X           X
```

Table 5. Contd.

(b) One of k, ℓ = 3 or 4

```
      PROB 4C    (K,L)  =  (1,3)
 T    T          T
 T    F          T
 T    D          T
 T    X          X
 F    T          F
 F    F          F
 F    D          F
 F    X          X
 D    T          D
 D    F          D
 D    D          D
 D    X          X    F    T
 X    T          X
 X    F          X-
 X    D          X
 X    X          X
```

```
      PROB 4C    (K,L)  =  (1,4)
 T    T          F
 T    F          F
 T    D          F
 T    X          X
 F    T          T
 F    F          T
 F    D          T
 F    X          X
 D    T          D
 D    F          D
 D    D          D
 D    X          X    F    T
 X    T          X
 X    F          X
 X    D          X
 X    X          X
```

(c) Both k and ℓ = 3, or 4

```
     PROB 4C   (K,L)  =  (3,3)
T    T          T
T    F          T
T    D          T
T    X          X
F    T          T
F    F          T
F    D          T
F    X          X
D    T          T
D    F          T
D    D          T
D    X          X
X    T          X
X    F          X
X    D          X
X    X          X
```

```
     PROB 4C   (K,L)  =  (3,4)
T    T          F
T    F          F
T    D          F
T    X          X
F    T          F
F    F          F
F    D          F
F    X          X
D    T          F
D    F          F
D    D          F
D    X          X
X    T          X
X    F          X
X    D          X
X    X          X
```

Table 6 contains the computer outputs for $\underset{\sim}{E}(\underset{\sim}{1}, \underset{\sim}{1})$, $\underset{\sim}{E}(\underset{\sim}{1}, \underset{\sim}{6})$ $\underset{\sim}{E}(\underset{\sim}{1}, \underset{\sim}{7})$ and $\underset{\sim}{E}(\underset{\sim}{7}, \underset{\sim}{7})$ for the QSTV2 implementation of QL-2 formulae analogous to (51a) and (51b). In this case, the full 8x8 table is given in a square array for QSTV (first table) and also for both (51a) and (51b) employing QSTV2 (second and third tables). Of these, the first table corresponds to the agreeing results using both (51a) and (51b), and also matrix multiplication, for QSTV It will be seen that the second and third tables for each (k, ℓ) agree with the first table for the first seven columns and first seven rows, in all the examples given. However, for the last row and/or last column, QSTV gives always X as output, while QSTV2 gives some entries, which are F in the case corresponding to (51a), and T in the case corresponding to (51b). The behaviour is closely similar to that observed in SNS. It is only necessary to replace the contradictory state, X, for the SNS terms, by the contradictory state, ϕ, for quantifier states.

It will be noticed that there are 2x3 = 6 entries when both $\underset{\sim}{k}$ and $\underset{\sim}{\ell}$ are not equal to 7 or 8, three entries if one of them equal to 7 or 8 while the other is not , and no differences at all between QSTV and QSTV2 if both are equal to 7 or 8.

Table 6. Check of equations similar to (51a,b) for $\underset{\sim}{E}(\underset{\sim}{k}, \underset{\sim}{\ell})$

PROB 8C (K,L) = (1,1) — 6 differences

	ALL	NAL	SOM	AON	NEX	EXS	IND	IMP
ALL	T	F	F	D	F	D	D	X
NAL	F	T	T	D	T	D	D	X
SOM	F	T	T	D	T	D	D	X
AON	D	D	D	D	D	D	D	X
NEX	F	T	T	D	T	D	D	X
EXS	D	D	D	D	D	D	D	X
IND	D	D	D	D	D	D	D	X
IMP	X	X	X	X	X	X	X	X

	ALL	NAL	SOM	AON	NEX	EXS	IND	IMP
ALL	T	F	F	D	F	D	D	X
NAL	F	T	T	D	T	D	D	X
SOM	F	T	T	D	T	D	D	X
AON	D	D	D	D	D	D	D	F
NEX	F	T	T	D	T	D	D	X
EXS	D	D	D	D	D	D	D	F
IND	D	D	D	D	D	D	D	F
IMP	X	X	X	F	X	F	F	X

	ALL	NAL	SUM	AUN	NEX	EXS	IND	I
ALL	T	F	F	D	F	D	D	X
NAL	F	T	T	D	T	D	D	X
SOM	F	T	T	D	T	D	D	X
AON	D	D	D	D	D	D	D	T
NEX	F	T	T	D	T	D	D	X
EXS	D	D	D	D	D	D	D	T
IND	D	D	D	D	D	D	D	T
IMP	X	X	X	T	X	T	T	X

PROB 8C (K,L) = (1,5) — 6 differences

	ALL	NAL	SOM	AON	NEX	EXS	IND	IMP
ALL	T	D	T	D	F	T	D	X
NAL	F	D	F	D	T	F	D	X
SOM	F	D	F	D	T	F	D	X
AON	D	D	D	D	D	D	D	X
NEX	F	D	F	D	T	F	D	X
EXS	D	D	D	D	D	D	D	X
IND	D	D	D	D	D	D	D	X
IMP	X	X	X	X	X	X	X	X

	ALL	NAL	SOM	AUN	NEX	EXS	IND	IMP
ALL	T	D	T	D	F	T	D	X
NAL	F	D	F	D	T	F	D	X
SOM	F	D	F	D	T	F	D	X
AON	D	D	D	D	D	D	D	F
NEX	F	D	F	D	T	F	D	X
EXS	D	D	D	D	D	D	D	F
IND	D	D	D	D	D	D	D	F
IMP	X	F	X	F	X	X	F	X

	ALL	NAL	SUM	AUN	NEX	EXS	IND	I
ALL	T	D	T	D	F	T	D	X
NAL	F	D	F	D	T	F	D	X
SOM	F	D	F	D	T	F	D	X
AON	D	D	D	D	D	D	D	T
NEX	F	D	F	D	T	F	D	X
EXS	D	D	D	D	D	D	D	T
IND	D	D	D	D	D	D	D	T
IMP	X	T	X	T	X	X	T	X

Table 6. Contd.

```
PROB 8C  (K,L) = (1,7)  — 3 differences

        ALL NAL SOM AON NEX EXS IND IMP    ALL NAL SOM AON NEX EXS IND IMP    ALL NAL SOM AON NEX EXS IND I

ALL      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

NAL      F   F   F   F   F   F   F   X      F   F   F   F   F   F   F   X      F   F   F   F   F   F   F   X

SOM      F   F   F   F   F   F   F   X      F   F   F   F   F   F   F   X      F   F   F   F   F   F   F   X

AON      D   D   D   D   D   D   D   X      D   D   D   D   D   D   D   F      D   D   D   D   D   D   D   T

NEX      F   F   F   F   F   F   F   X      F   F   F   F   F   F   F   X      F   F   F   F   F   F   F   X

EXS      D   D   D   D   D   D   D   X      D   D   D   D   D   D   D   F      D   D   D   D   D   D   D   T

IND      D   D   D   D   D   D   D   X      D   D   D   D   D   D   D   F      D   D   D   D   D   D   D   T

IMP      X   X   X   X   X   X   X   X      X   X   X   X   X   X   X   X      X   X   X   X   X   X   X   X


ROB 8C  (K,L) = (7,7)  — No differences

        ALL NAL SOM AON NEX EXS IND IMP    ALL NAL SOM AON NEX EXS IND IMP    ALL NAL SOM AON NEX EXS IND IMP

 L       T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

 L       T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

 M       T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

 N       T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

 X       T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

 S       T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

 D       T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X      T   T   T   T   T   T   T   X

 P       X   X   X   X   X   X   X   X      X   X   X   X   X   X   X   X      X   X   X   X   X   X   X   X
```

This is closely similar to the behaviours in SNS as mentioned for Table 5. In fact, it is readily verified that the differences between QSTV and QSTV2 are produced only if

$$\underline{\underline{t}}(\underset{\sim}{a} \,/\, \underset{\sim}{k}) = D \quad \text{and} \quad \underset{\sim}{b} = \emptyset,$$

or

$$\underline{\underline{t}}(\underset{\sim}{b} \,/\, \underset{\sim}{\ell}) = D \quad \text{and} \quad \underset{\sim}{a} = \emptyset \qquad (52)$$

The corresponding formulae for SNS can be put in the same pattern as

$$\underline{\underline{t}}(\underline{\underline{a}} \,|\, s(k)) = D \quad \text{and} \quad \underline{\underline{b}} = X$$

or

$$\underline{\underline{t}}(\underline{\underline{b}} \,|\, s(k)) = D \quad \text{and} \quad \underline{\underline{a}} = X \qquad (53)$$

These results are readily derivable from BVMF algebra, and this is left to the reader who may prove them even more generally for GLOGIC, after reading MR-65.

Appendix

The appendix to Part III, MR-62 has been referred to in MR-60 and 61. These references deal essentially with Problems 4, 6, 7 and 8. In some cases the reference is made to MR-65 for the same problems. Since the theoretical discussions have to be given in some detail, no appendix is added here, but both the materials intended for this appendix as well as for MR-65 are given in some detail in the report MR-65 forming Part IV of MATLOG series.

The proposed detailed treatment of QL-1B, mentioned in page 62, MR-61, and of tensor relations and GLOGIC, mentioned in MR-62 at various places, have not been written up in MR-65 for lack of technical assistance.

STEREOCHEMISTRY OF COLLAGEN

(Invited Review Article to appear in
Intl. J. Peptide and Protein Res.)

G.N. Ramachandran
INSA Albert Einstein Professor
Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

Matphil Reports No. 63, May 1987

Stereochemistry of Collagen

G.N. Ramachandran

Mathematical Philosophy Group
Indian Institute of Science
Bangalore, India 560 012

ABSTRACT

This review article, based on a lecture delivered in Madras in 1985, is an account of the author's experience in the working out of the molecular structure and conformation of the collagen triple-helix over the years 1952 - 1978. It starts with the first proposal of the correct triple-helix in 1954, but with three residues per turn, which was later refined in 1955 into a coiled-coil structure with approximately 3.3 residues per turn. The structure readily fitted proline and hydroxyproline residues and demanded glycine as every third residue in each of the three chains. The controversy regarding the number of hydrogen bonds per

The author is deeply grateful to Prof. C.H. Li, for having kindly invited him to contribute his review article based on the lecture delivered in the Central Leather Research Institute (CLRI), Madras, in July 1985. The sequence here closely follows that of the lecture, but the text has been edited so as to be more suitable for this journal. The figures have been pruned and captions added, along with the more pertinent references to the text.

tripeptide could not be resolved by x-ray diffraction or energy minimization, but physicochemical data, obtained in other laboratories during 1961-1965, strongly pointed to two hydrogen bonds, as suggested by the author. However, it was felt that the structure with one straight NH ... O bond was better. A reconciliation of the two was obtained in Chicago in 1968, by showing that the second hydrogen bond is via a water molecule, which makes it weaker, as found in the physicochemical studies mentioned above. This water molecule was also shown, in 1973, to take part in further cross-linking hydrogen bonds with the OH group of hydroxyproline, which occurred always in the location previous to glycine, and is at the right distance from the water. Thus, almost all features of the primary structure, x-ray pattern, optical and hydrodynamic data, and the role of hydroxyproline in stabilising the triple helical structure, have all been satisfactorily accounted for. These also lead to a confirmation of Pauling's theory that vitamin C improves immunity to deseases, as explained in the last section.

Stereochemistry of Collagen

1. Introduction

It was very kind of Dr. Thyagarajan, Director of CLRI, to have invited me to inaugurate the renewed activities in the field of basic sciences in this Institute. I am particularly happy about this since I have been associated with CLRI right from its inception in the 1950's. I could not think of a better topic than the 'Stereochemistry of Collagen' for the inaugural address, since it has been one of the main fields of activity, in our laboratories of Molecular Biophysics, both in the University of Madras, and in the Indian Institute of Science, Bangalore. As I have not been doing original work in this field for the past few years, I propose to give a short account of the various developments in this subject made in our laboratories, with special reference to the molecular structure of collagen. Starting from x-ray diffraction studies in the early 1950's and the proposal of the correct triple helix for collagen in 1954-55, which were followed up by conformational analysis and energy minimization for its refinement, they have led to the delineation of the role of hydroxyproline in collagen in

the early 1970's and the linking up of this to the role of vitamin C in immune response in 1978. These are broadly reviewed in this lecture.

2. Protein helices

Stable structures of protein molecules depend upon two aspects, namely the unbranched peptide chain of the protein molecule and the occurrence of hydrogen bonds between different parts of the protein chain that stabilizes its secondary structure. The importance of hydrogen bonds for helix formation became established with the theoretical elucidation of the now-well-known α-helix by Pauling in 1951. Once the parameters of the helix were found from stereochemistry, it took no time at all to prove the correctness of these by x-ray diffraction, as was done by Perutz. Within a year or two of the demonstration of the α-helix in the KMEF group of fibrous proteins, came the even more exciting double helix for DNA as proposed by Watson and Crick in 1952. The helix era had begun.

I mention all these because I was encouraged to enter this field by reading the beautiful series of papers

published by Pauling and coworkers in 1951, and, when I was appointed Professor and Head of the newly started Physics Department in Madras in 1952, I chose x-ray diffraction, and x-ray crystallography in particular, as the main theme of our laboratory, and their application to biomolecules as the main aspect of this field that is to be pursued vigorously. However, I did not know where to begin, and which molecules, or biopolymers, were the most profitable ones to study. This problem was resolved by the happy coincidence of a visit by Prof. J.D. Bernal to Madras in the early 50's. When I put this question to him, he told me that he was not very happy with the various structures of collagen that had been proposed in the literature at that time, and that the problem was wide open. Even more than that, he indicated that there were some specimens of shark fin ray collagen (elastoidin) in the Department of Biochemistry of Madras itself.

3. First contacts with collegen via CLRI

This helped us in starting our studies. But this material was not the best one suited for x-ray diffraction purposes. We wanted kangaroo tail tendon (KTT) or beef

Achilles tendon. Here again, it was very lucky that the Central Leather Research Institute was in Madras; in fact it was our neighbour in Adyar. I contacted Dr. Nayudamma in CLRI and they obtained for us a big tubeful of KTT from Australia. For beef tendon, the usual procedures of obtaining the pure protein was done in CLRI itself. The availabality of x-ray diffraction photographs in our own laboratory for first hand studies was very useful in the solution of the correct structure, as we shall see later.

Coming to the story of the progressive steps in obtaining the detailed molecular architecture (secondary structure) of the collagen helix, I shall only touch upon the highlights, particularly with reference to the work done in our laboratories. This report is not intended to be a review, or an authoritative account, but only a reminiscence of the studies made in Madras, Chicago and Bangalore.*

*Reference may be made to the reviews (1-4) by the author, and, therefore, this lecture will not cite full references to the literature, but only the vital ones.

The most important feature of collagen, that has relevance to the molecular architecture of its protofibril, is the characteristic amino acid composition of the protein, which had been determined at that time for a wide range of biological sources (Fig. 1). It is widely different from that of other proteins, The most important feature is the existence of one-third the number of amino-acid residues as glycine, and the existence of almost another one-third of the residues as the imino-acid residues proline and hydroxyproline. The positively and negatively charged amino acids are few, but occur in approximately equal amounts. The other features that had been reported in the literature were the data on infra red dichroism of the fibre and optical rotation of the solution. The infra red data showed that the groups N — H and C = O, of the backbone of the protein chain, are approximately at right angles to the fibre axis, and the specific rotation indicated that the helix was left handed, having a sense apposite to that of the α -helix. The infra red data also indicated that all the peptide units were trans, and that no cis residues were present. The x-ray pattern did not have too much of data, and had been interpreted diffierently by different workers.

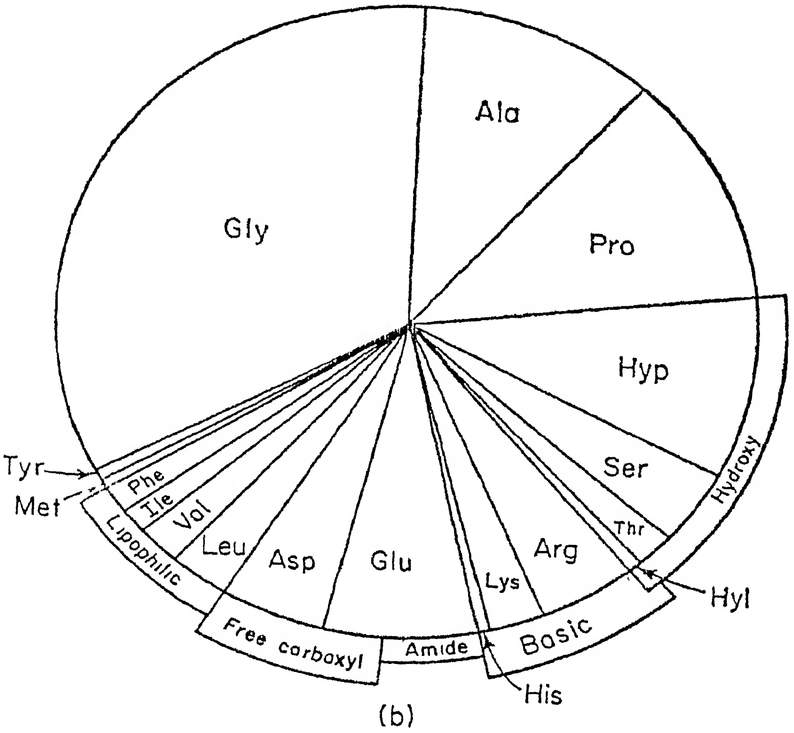


Fig.1. Amino acid composition of collagen. Note the occurrence of 1/3 the number of residues as glycine and practically another 1/3 as the imino-acid residues proline and hydroxyproline, having rigid side chains which impose a left-handed twist for the helix (see below). (From the book (1)).

There were several proposals in the literature, for the chain configuration in collager some of which are shown in Fig. 2(a-d). As will be seen from these figures, the nature of the helix proposed by different workers widely differred from one another, , and the one shown in Fig.2(c) was not helical at all. Some had cis peptide units in addition to the trans peptide units, a feature that was contrary to the infra red information that only trans residues were present. It must be mentioned that there was no special feature in any of them, which required that one-third of the residues must be glycine.

4. Proposal of the first structure in 1954

I had the good fortune of having a trained crystallographer and an excellent scientist in Dr. Gopinath Kartha, who joined me as a postdoctoral worker, in our studies on collagen. This gave a great spurt to our activities. Kartha and I made various attempts at building a structure that will fit the x-ray diffraction data, in addition to all the chemical and physico-chemical data mentioned above. A fresh indexing of the x-ray pattern (see Fig. 5), indicated that the protofibril was about 12Å in diameter, and that it had a repeat along the

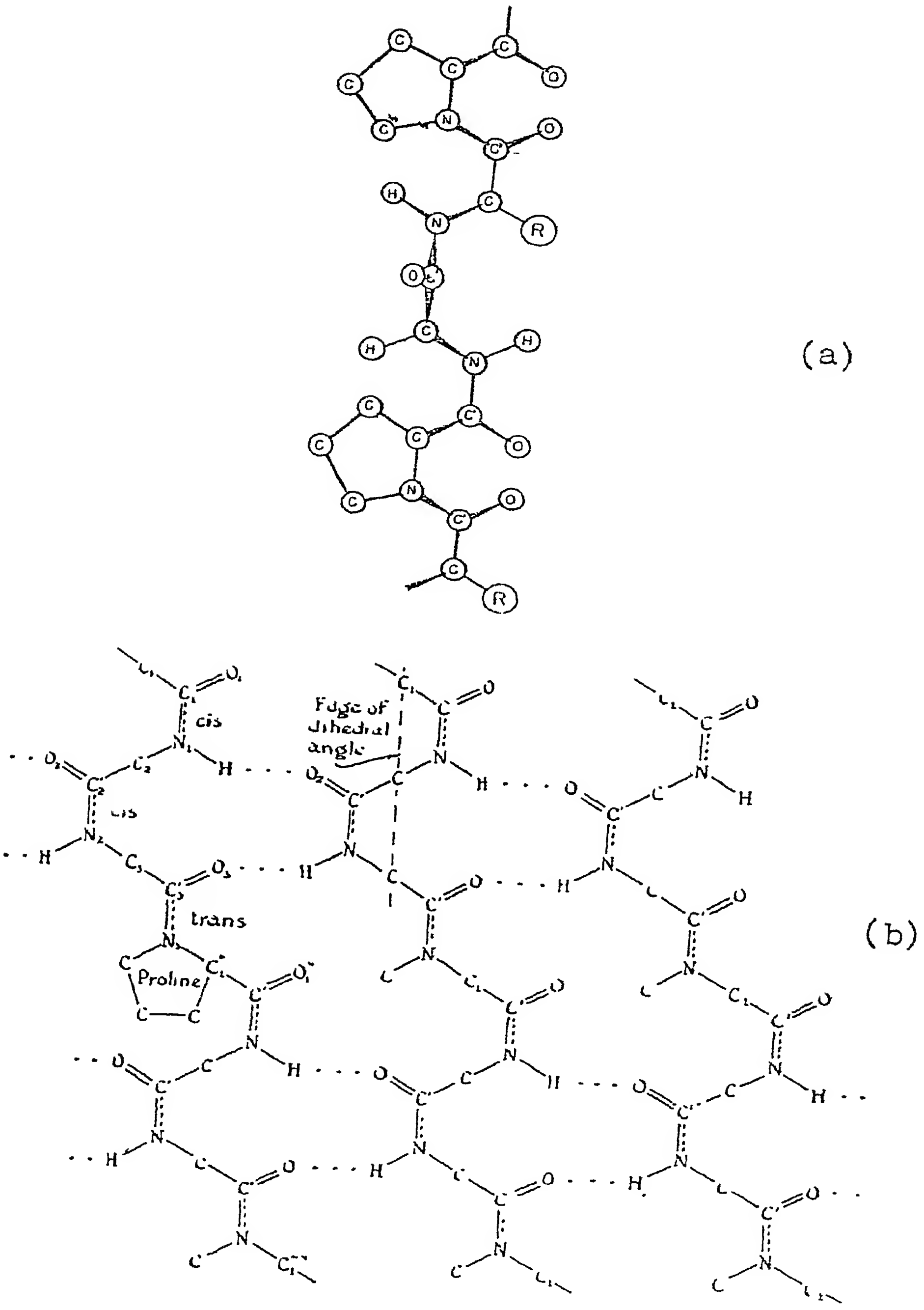


Fig.2. Diagrams illustrating the structures proposed for collagen by (a) Astbury and Reid (1940), (b) Pauling and Corey (1951), (c) Randal and coworkers (1952), and (d) Crick (1954). Note that the structures in (a), (b) and (d) employ <u>cis</u> residues for proline, while (c), which is all <u>trans</u>, is not helical at all. None of them demand that glycine should form one-third the number of residues. (From (2)).

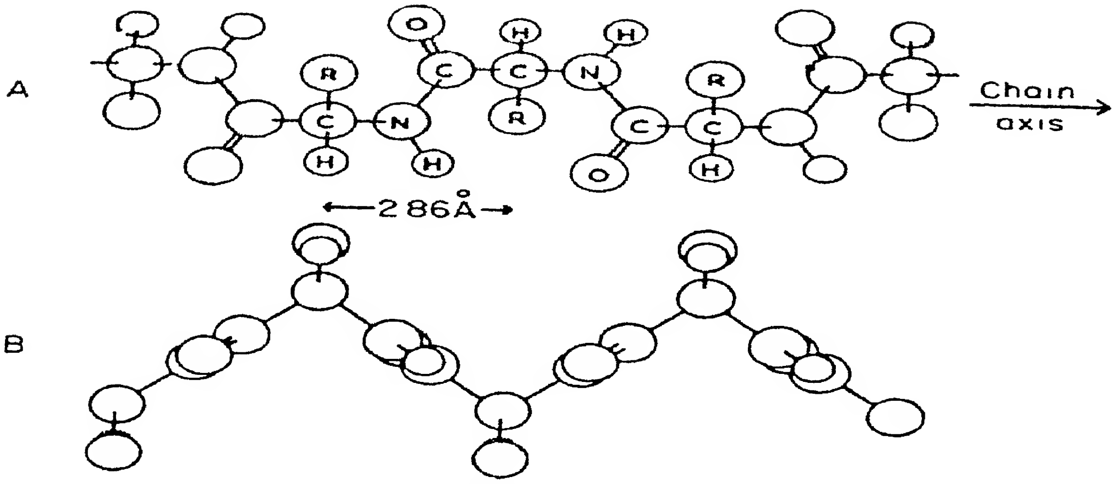


FIGURE 9(c)

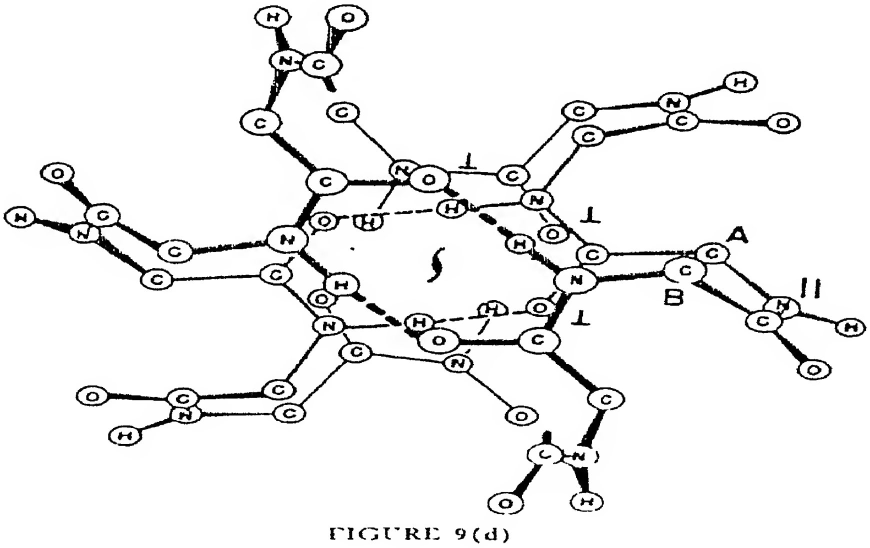


FIGURE 9(d)

Fig.2. Continued

fibre axis of approximately 9-10Å with probably three residues in the repeat, corrêsponding to the meridional spacing of 3Å, and having three such chains in the unit cell as indicated by the density. Kartha and I, being both crystallographers, gave the greatest value to the diffraction pattern and started building structures which were left-handed helices, as required by optical rotation, and which had N—H and C = O bonds at right angles to the helix axis, as required by infra red data. We also put in the condition that no cis residues were present, as had been found also from infra red data. All of these were conditions that were happily chosen, and found to be even in the final model.

But the most interesting feature was that the occurrence of proline and hydroxyproline in the L-configuration in collagen restricts the freedom of orientation of the chain in relation to the fibre axis and permits only a left-handed helix. We, therefore, assumed that each individual chain had a helical configuration of three residues per turn, with the crystallographic symmetry 3_2. When three such chains, each having the left-handed 3-fold screw axis symmetry, were put together, and hydrogen bonds were fitted in, we found that each of them was again related to the other two by a similar 3-fold screw axis. A perspective view of the structure so obtained,

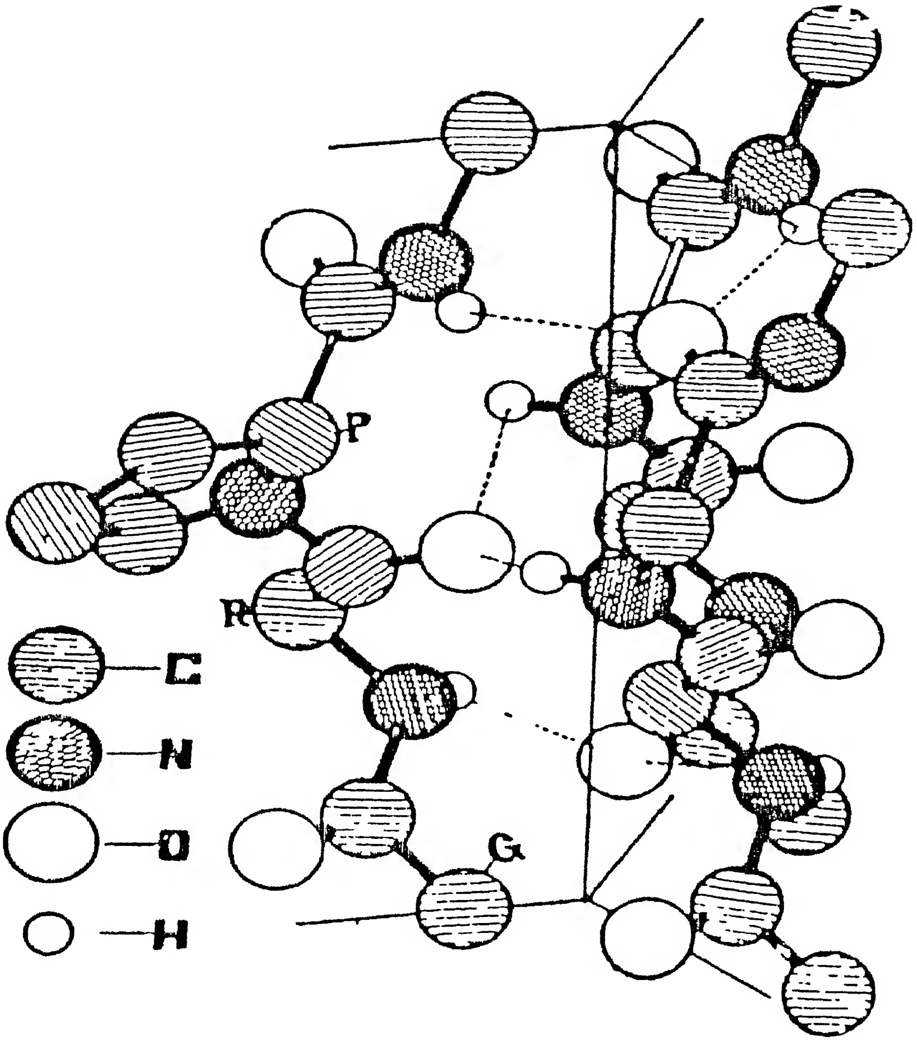


Fig.3. Perspective view, from the side, of the contents of one unit cell, for a height of 9— 10Å, of the collagen triple helical structure (5). Note that the hydrogen bonds are nearly at right angles to the helical axis, all residues are trans, and that L-proline side chains readily fit in the structure, giving it a left-handed twist for the helix. (From (5)).

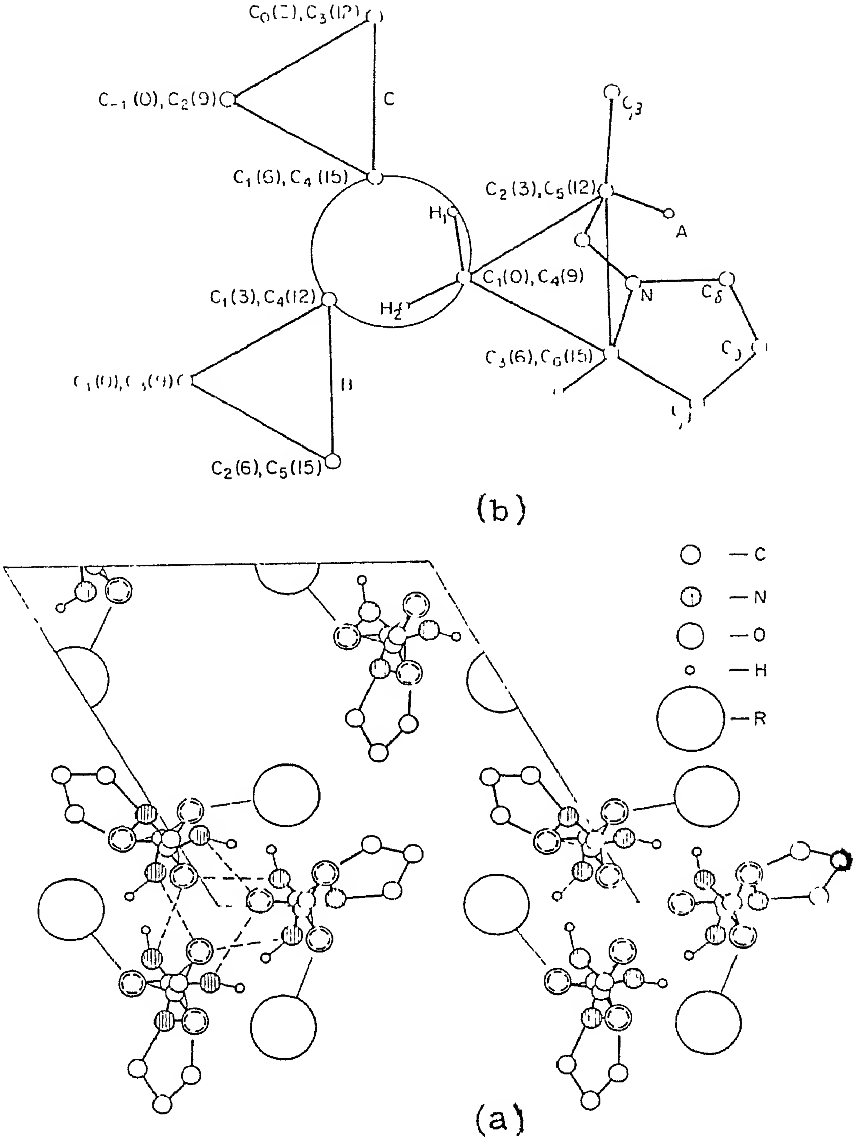


Fig.4. The structure in Fig.3 projected down the helical axis — (a) triple helices packed in an hexagonal unit cell. Only the hydrogens in N — H groups are shown. Note the feature that one of the three α-carbon atoms in the three residues occurring for one full turn cannot have any side-chain β-carbon atom, while the other two can have any side chain attached to them, including especially proline. This feature is particularly clear from (b) where the two hydrogen atoms of glycine attached to C^{α} , and the C^{β}-atoms of the side chain in the other two residues, are also marked in one of the chains. (From (6)).

(which we published in Nature (5) in 1954) is shown in Fig. 3. Fig. 4(a) contains a projection down the helical axis of the proposed crystal structure (reproduced from (6)).

As will be seen from Fig.3, the hydrogen bonds are nearly at right angles to the helical axis and the peptide units are all in the *trans* configuration. Also the proline side chain comes on the outside of the triple-chain protofibril. This is also seen very clearly in Fig. 4(a). However the most significant feature of the molecular structure that automatically came out of the analysis was that every third residue in each of the three chains must be glycine. The essence of this is shown in Fig. 4(b), where it will be seen that every third residue cannot accommodate the β-carbon of any of the other nineteen amino acids if inter-chain hydrogen bonds are to be made in the protofibril, as in Fig.4(a). Therefore, the experimentally observed composition of one-third glycine, which is a very characteristic feature of collagen, comes out as a fundamental feature required for the molecular fit of the three helical chains with one another in the triple helix. It is also seen from Fig. 4(a), that proline or hydroproline could redily occur in the other two

locations between two glycine residues, in the sequence —Gly—X—Y—, of the molecular chain.

It is interesting that the exact 3-fold screw symmetry, of the chain invoked here, (but modified to a non-integral screw axis with 10 residues in 3 turns later) was found very soon to be valid for the polypeptides poly-L-proline (7) and poly glycine (8), and later / (Sasisekharan, 1959), for poly-L-hydroxyproline also, all of which have a unit height of about 3Å.

5. Supercoiled structure with $3\frac{1}{3}$ units per repeat (9,10)

On careful examination, we were not fully satisfied with this structure, although it was apparently a good first approximation to the correct one. The most important feature that had to be refined was the number of residues per turn in the helix. Fortunately, just two years earlier, Cochran, Crick and Vand(1952) had published a thorough analysis of the theoretically expected diffraction pattern of helical structures, and how the geometrical features relating to the structure, such as unit twist, unit height, and pitch, could be obtained from a measurement of the diffraction pattern. Therefore G.K.Ambadi of our department took a diffraction

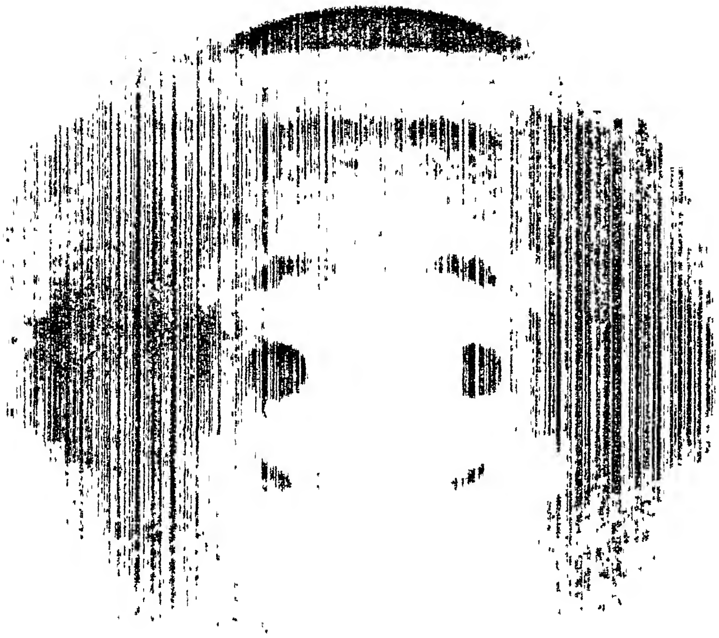


Fig.5. X-ray diffraction pattern of collagen obtained by keeping the fibre at a suitable angle to the x-ray beam for making the 3Å spot meridional. Apart from substantiating the approximate value 10/3 for the number of residues per turn (n), it also helped in getting the more accurate value of 3.27 for this quantity obtained later in 1960.

picture of KTT, kept wet and fully stretched, and inclined to the x-ray beam, so as to record all the reflections down to 1.5Å. When we analysed this pattern (Fig. 5) according to the formulae of Cochran _et al_[9], we found that the correct value, for the number of residues per turn in the collagen helix, is close to 3.3, or 10/3 (the nearest ratio of small integers), rather than 3. Therefore, the individual chains cannot be helices with an integral number of units per turn. The schematic picture of one such helix, having about 3.3 residues per turn, is shown in Fig. 6(a). In this, P_1, P_2, P_3 etc., represent the C^{α} atoms of adjacent peptide units and these are separated by a twist per residue of 110° about the helical axis, and a displacement of 3Å at right angles to the diagram. The number of residues per turn (n) is therefore 360°/110° = 3.27. This value, rather than the value of 10/3 = 3.33, is chosen for Fig. 6 because it is based on a more accurate determination made in 1960. If n = 10/3, then the unit twist will be 108° instead of 110°; both of these are to be compared to the value of 120° corresponding to n = 3 as in Fig. 4(b).

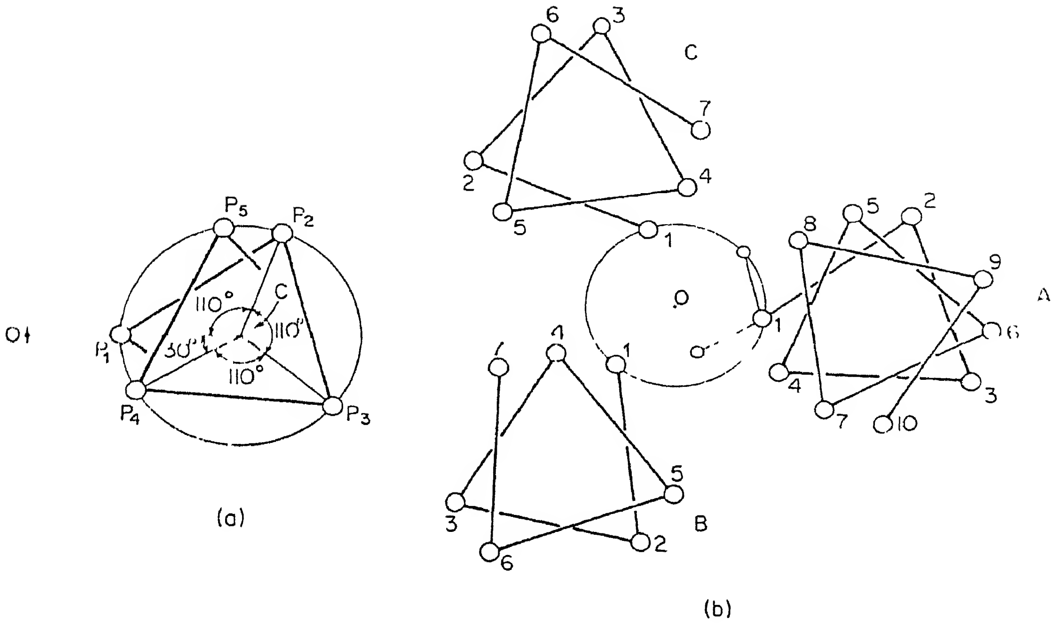


Fig.6(a) Projection of the backbone of a non-integral helix with approximately ten residues in three turns. (This value corresponds to 108° for the unit twist. However, the more accurate value of 110° corresponding to $n = 3.27$, which was found later in 1960 in our laboratory, is used in this figure.)

(b) When three such helices are put together, as in Fig.4(b), it is seen that the condition that every third α-carbon atom must be on the inside of the triple helix, is lost — for example, for the α-carbon atoms 4, 7 and 10.

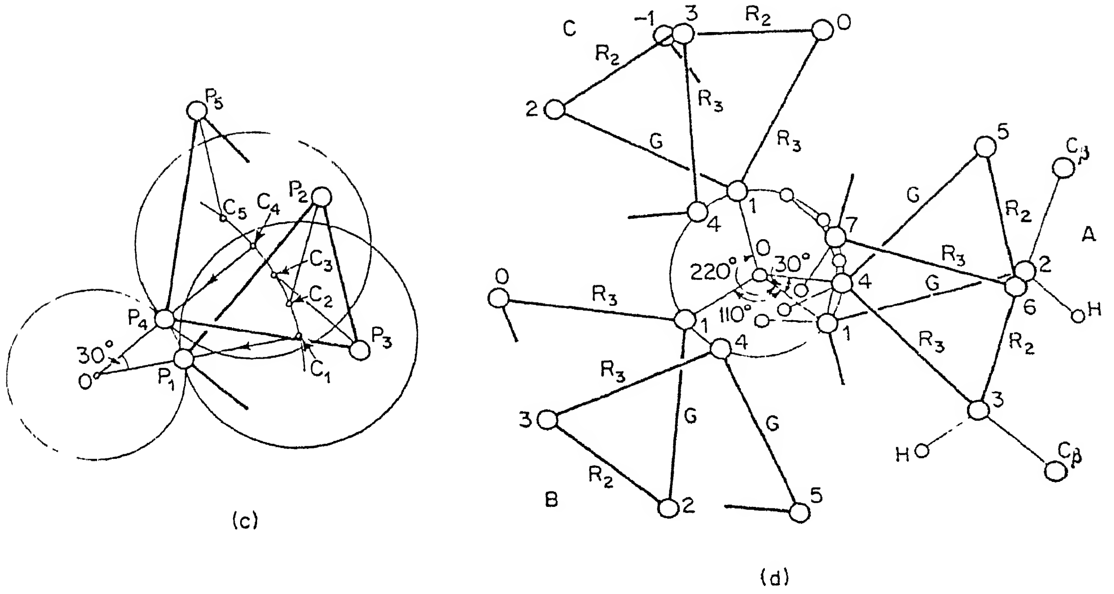


Fig.6(c) This condition can be restored by giving a right-handed superhelical twist of 30° (360° - 3 x 110°) for the indiviaual helices as shown in this figure.

(d) The pattern of the molecular arrangement employing three such supercoiled left-handed minor helices, all of which are wound about the central axis with a right-handed twist. The value of 30° for three residues of the supercoiling twist will lead to a major helix pitch of about 108Å, corresponding to 36 residues. (From (1)).

However, when three such helices are put together in the pattern of Fig. 4(b), as shown in Fig. 6(b), the condition that every third α-carbon atom mut belong to glycine, is lost. It took Kartha and me some time to figure out how to overcome this difficulty. Finally, we discovered a way, and this was to make the three helices intertwine themselves further by twisting around the common central axis of the structure, so as to form what is called a "coiled-coil structure". This feature is shown for a single chain in Fig. 6(c). A glycine α-carbon atom P_1, and the next such atom P_4, both now occur at the same distance from the centre O and on the inside of the major helix. However, they are not one above the other as in Fig. 4(b), but are relatively displaced by a right-handed twist of 30°. The origin of this value, 30°, for the superhelical twist will become clear from Fig. 6(d), where three left-handed helical chains are put together and wound about the central axis. In this case, the atom C_1^α of one chain (A), and that of the next one (B) to the left of it, are separated by 110° about the major helical axis, and this gives the value of the number of units per turn of 3.27 as required by x-ray data. If we take the C_1^α atoms of

of the chains A, B and C, they are at heights of 0 Å for A, about 3Å for B and about 6Å for C. Then, we come back to the corresponding C^{α} atom, C^{α}_{4}, of chain A at about 9Å. We have effectively made a total left-handed twist of 3 x 110°, and therefore we are left with a right-handed twist of 30° between C^{α}_{1} and C^{α}_{4} of the same chain A.

It is obvious from symmetry considerations that this pattern can be repeated in chains B and C also and the resultant structure then has a non-crystallographic helical true symmetry with a unit twist of —110° (the minum refers to left-handed), about the common central axis, and has a repeating unit consisting of three residues (—Gly—X—Y—) in each of the chains. This is shown in Fig. 6(d). Consequently each peptide chain has 36 residues in the pitch of the triple helix. (This will be 30 if $n = 3\frac{1}{3}$.) A side view of the molecular structure so obtained is shown in Fig. 7.

This structure was also published in Nature (10) in 1955, and the projection down the fibre axis of this structure is shown in Fig. 8(a,b). It will be seen from this that all the stereochemical properties of the non-coiled-coil structure — namely that the hydrogen bonds are nearly at right angles to

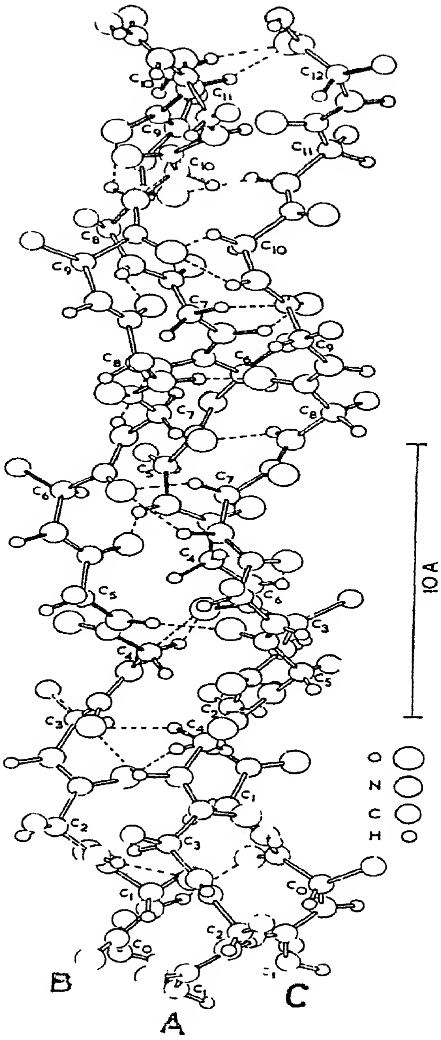


Fig.7. Perspective view of the supercoiled collagen structure for a height of about 30Å, showing the backbone atoms. (From (1)).

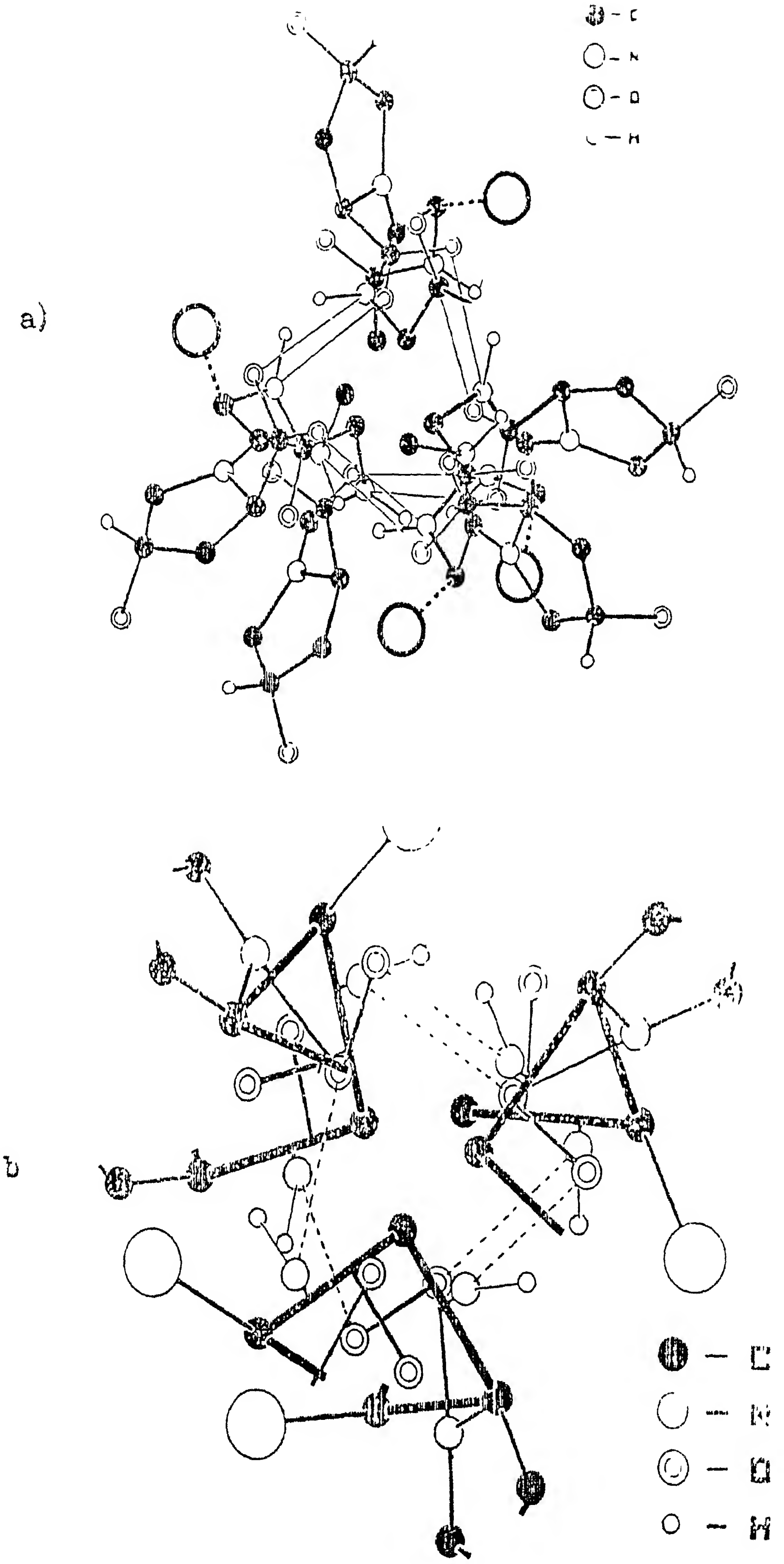


Fig.8(a) Projection of the atoms, in the backbone and proline groups, in a height of three residues from 0 to 8.6 Å along the c-axis, drawn analogous to Fig.4(a). Note that all the features pointed out for the non-supercoiled triple helix in Fig.4(a) are preserved in this also.

(b) This diagram brings out the path of the backbone peptide units (which is seen to follow the pattern in Fig. 6(d). In addition to α-carbon atoms, N—H and C—O groups involved in hydrogen bonding are also shown. (From (6)).

the fibre axis, that every residue is a planar trans peptide unit, that the individual helices are left-handed, and that Pro and Hyp residues can be readily incorporated, in addition to the stereochemical requirement that Gly should occur once in every three residues in each peptide chain — are still preserved in the supercoiled structure. As will be seen from Fig. 7, the supercoiling about the central axis of the protofibril is quite small, being only approximately 90° for the whole length of the molecule that is shown in the figure.

By this time, the subject of the triple helical structure of collagen had drawn the attention of two other laboratories in Britain (Rich and Crick (11) in Cambridge and Cown et al (12) in London) who had also examined the collagen structure on the basis of their observations on poly-glycine (8) and poly-L-proline (7) respectively. They obtained essentially the same conclusions as we did, namely that a detailed fitting with the spacings, of the layer lines in the observed x-ray pattern of collagen, requires that the structure must be built of three individual left-handed helices, with right-handed

supercoiling, as shown in Fig. 8(b). They also used a value of 10/3 for n, the same as what was adopted by us in 1954-55.

There were minor differences in the details as between the proposals from the three laboratories (see the next section), but the basic features were the same in all of these. That this structure is essentially correct is shown by the optical diffractometer pattern (Fig. 9) taken in 1955 by Dr. Cowan in King's College , London, using a replica of our molecular structure shown in Fig. 8(a), and employing diffraction of light, instead of x-rays, for recording it. It will be seen from Fig.9 that this pattern exhibits all the features of the x-ray pattern, and contains the equivalents, of the meridional spacing of 3Å (10th layer), and two other strong non-meridional layers (3rd and 7th), in addition to a weak 4th layer. In fact the x-ray photograph is almost completely duplicated by this optical diffractograph. Thus, the essential features of the helical structure, and its supercoiling, become well established by the end of 1955.

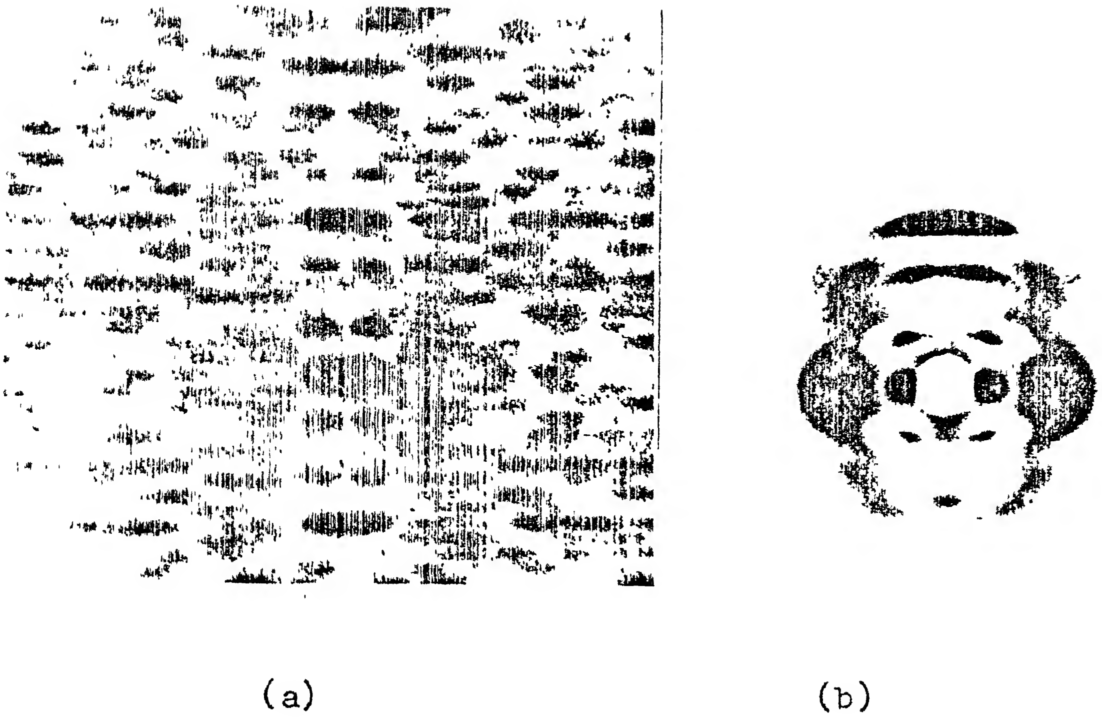


Fig.9 (a) Optical transform recorded using the supercoiled triple helix.

(b) Diffraction pattern of collagen fibre kept at 75° to the x-ray beam. Note the almost exact correspondence, in the central region, of the top halves of the two figures. (In the x-ray pattern the intensity falls off rapidly with the distance away from the centre, owing to disorder.) (From (6)).

6. The question of one or two hydrogen bonds per three residues

Although there was thus general agreement as to the helical parameters associated with the collagen triple helix, there were minor differences in the molecular structure as proposed by the three laboratories mentioned above. The main point of difference was, as to whether the number of hydrogen bonds stabilizing the structure per three residues, was one or two. Pauling had enunciated the principle that the maximum number of hydrogen bonds would be made in a stable structure, and we had therefore tried to introduce both the NH ... O hydrogen bonds that are possible between adjacent protein chains, while the third NH was nowhere near the correct orientation for a hydrogen bond to be formed. Rich and Crick (11) and Cowan, McGavin and North (12) had both concluded that only one strong hydrogen bond is possible, if the structure is to be not too closely packed, and, in their structures, the second NH group that could be involved in a hydrogen bond was left free. When the preprints of these papers reached us, we also looked into this question, and it was found that an one-bonded structure could be built (13). However, this could be

done by rotating the minor helices either in an anti-clockwise direction by +40°, or a clockwise direction by -15°, from the double-bonded structure. Both Rich and Crick and Cowan et al preferred the former (plus) structure, but we found that the latter was superior (Only this minus structure was used for later studies in our laboratory).

During the succeeding years, the structure was looked at in great detail in Madras, and the relative merits of the one-bonded and two-bonded structures, both from considerations of molecular packing, and of their agreement with the x-ray diffraction pattern, were examined in our laboratory. The results were presented in an International Symposium on Collagen arranged by CLRI in 1960 (14). The main conclusion was that the best conformation that could be worked out for both our (minus) one-bonded structure and our two-bonded structure were equally good; but we preferred the two-bonded structure because of the possible stability introduced by the additional hydrogen bond. (In fact, these studies, made mainly in collaboration with V. Sasisekharan and C. Ramakrishnan, brought home to us the fact that there were no hard and fast criteria available in the literature for judging a structure to be good or bad,

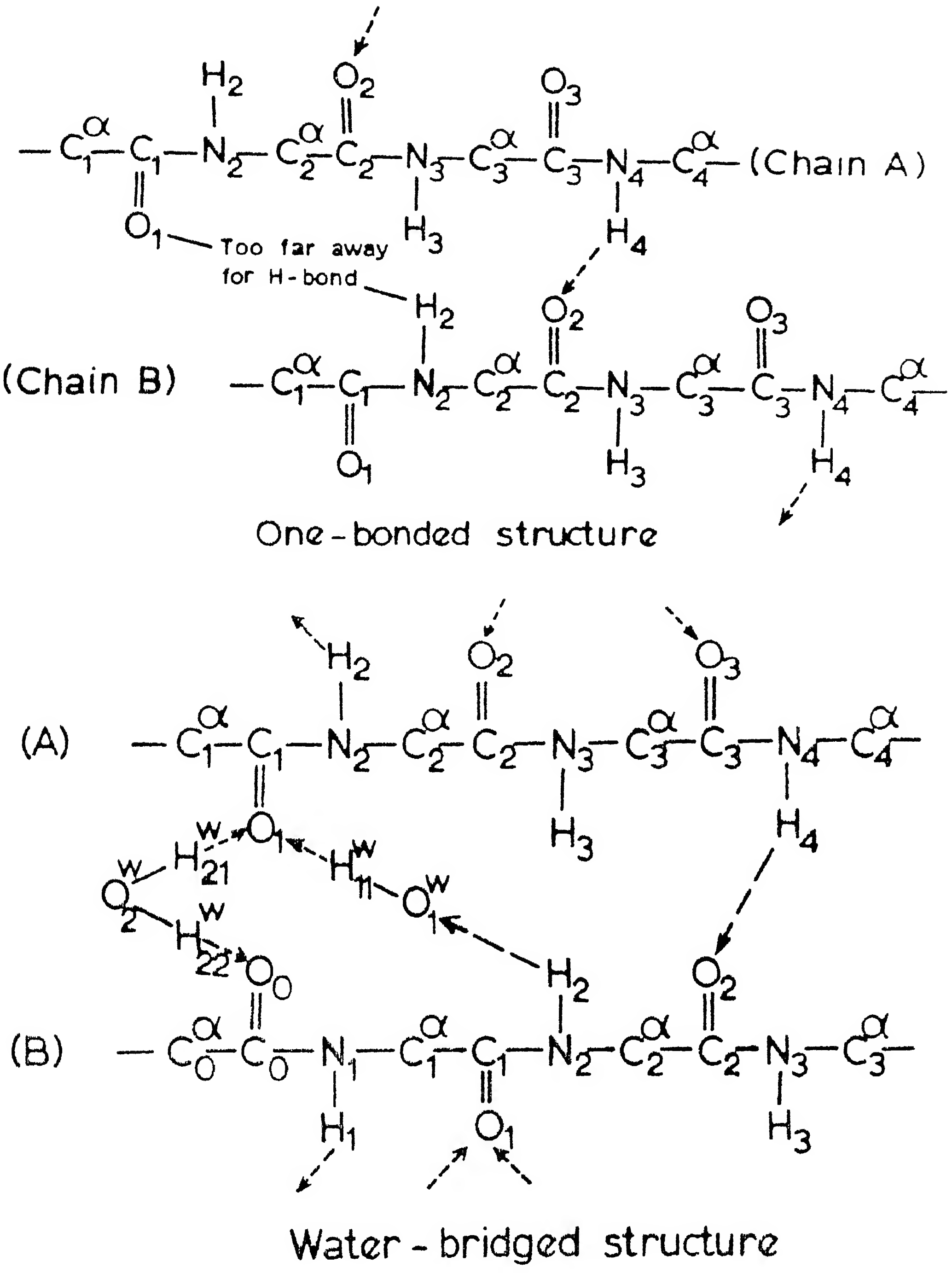
and this led us to our studies on protein conformation and on stereochemical criteria for peptide structures in general, but that is another story.)

From the point of view of the x-ray diffraction data, the calculated Fourier transforms of the two structures were very close to one another, and both agreed fairly well with the observed intensity distribution in the layer lines. In fact, independently, Rich and Crick (15) also reported good agreement with x-ray data of an one-bonded structure close to our minus structure.

Attempts were therefore made to resolve the question by means of conformational energy calculations of the two structures (16). Although these gave, as expected, a lower energy for the structure with two hydrogen bonds per three residues, it was found to be only marginally superior, because the extra stabilizing energy was mainly contributed only by the additional hydrogen bond, and the *non-bonded energy* was, actually, *inferior* for the two-bonded structure. The situation was puzzling.

In fact, the question of the number of NH ... O hydrogen bonds per tripeptide in the collagen structure was investigated experimentally in several laboratories all over the world during the 1960's, using various physicochemical techniques. Thus, data from widely different methods, such as deuterium exchange (Bensusan and Nielsen, 1964), tritium exchange (Englander and Von Hippel, 1967 and McBride and Harrington 1964 calculation of shrinkage and denaturation temperatures (Harrington, 1964), all indicated the existence of two NH ... O hydrogen bonds per three residues, rather than one (for a full account see (17)).

Although there was all this evidence in favour of the two-bonded structure, and although, as will be seen from Fig. 10(a), there is a gap in the cross linking bonds between the chains in the protofibril of the triple helix in the one-bonded structure, I had an intuitive feeling that the structure with two straight interchain NH ... O hydrogen bonds may not be the final answer, but that there must be some other way of building the hydrogen bonds for the NH groups in a structure having the chain configuration close to that of the



One-bonded structure

Water-bridged structure

ig.10(a): Diagram showing the only set of hydrogen bonds of the type N_4H_4 ... O_2 linking the backbones of neighbouring peptide chains in the collagen triple helix. Note that these are too few, and are not sufficient to produce good stability for the structure.

(b): On introducing two water molecules with O_1^W and O_2^W, several OH ... O linkages are produced, in addition to a second NH ... O hydrogen bond of the type N_2H_2 ... O_1^W. Note that practically every NH and every CO is involved in the hydrogen bond network. (Reproduced from (3)).

one-bonded structure. This idea was supported by the observation that all the techniques mentioned above indicated the existence of one strong hydrogen bond and one relatively weaker hydrogen bond for every three residues — the weakness and strength being indicated by the ease with which the hydrogen atoms in the NH groups are displaced by deuterium, or tritium, as the case may be.

Till about 1967, there was no clear resolution from theory of this problem. However, a study made in Chicago along with R.Chandrashekaran gave us a very satisfactory solution (18). This was that the strong NH ... O hydrogen bond is a direct one (N_4 H_4 ... O_2) between the backbone atoms of neighbouring chains, while the second weaker one is from the free NH group (N_2H_2) of Fig.10(a) to the oxygen O_1^W of a water molecule which again links up with the oxygen of a CO group (C_1O_1) in a neighbouring chain, via on OH ... O bond (Fig. 10(b)). Then, as will be seen from Fig. 10(b), it is also possible to have one more water molecule, ($O_2^W(H_{21}^W$, H_{22}^W)), both of whose hydrogens bond with CO groups of neighbouring peptide chains. All this is seen in a wire model of the collagen structure with water bridges shown in Fig. 11.

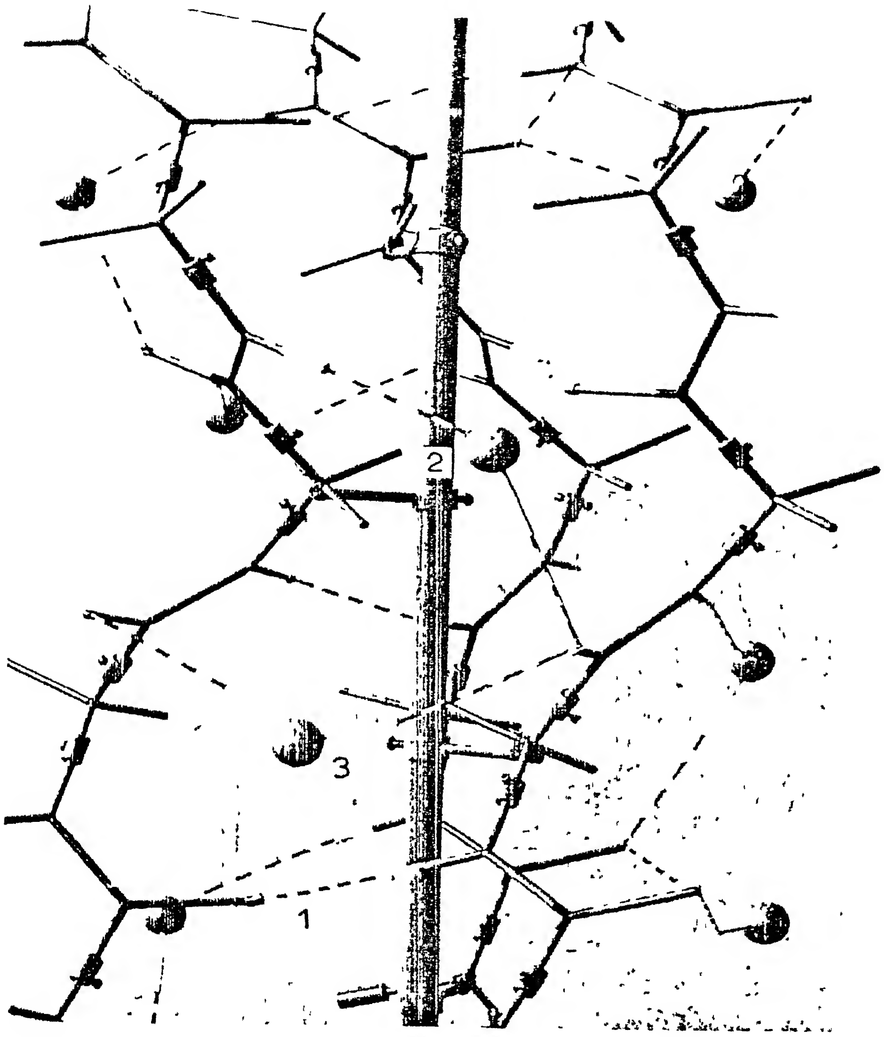


Fig. 11 : Photograph of a wire model showing the hydrogen bond cross links in the water bridge structure. The rods represent the backbone bonds and the balls water oxygens. Hydrogen bonds are indicated by striped wires.

(Reproduced from (18)).

Water molecules are well-known to be firmly bound to protein structures, but this feature had not been properly taken into account previously for collagen, although Cowan et al (12) had suggested it. Soon after, in 1969, Yonath and Traub (19) also cameout with possible locations for bound water molecules in the collagen structure based on their refinement of the structure of poly(Gly—Pro—Pro), which is relevant for the portions of the collagen triple helix having this sequence.(see also (20)).

However, this was not the whole story, for the same water molecule involved in the NH ... O bond in Fig. 10 (b) could also be shown to serve the purpose of forming additional hydrogen bonds with the hydroxyl group of hydroxyproline side chains, producing still greater stability to the triple chain. This is discussed below.

7. Hydroxyproline and its role in the collagen structure

It should be mentioned that vigorous studies, of a physico-chemical and biochemical nature, had been going on in various laboratories during the 50's and 60's, and definite evidence that the collagen molecule is built up of three chains, and that they form a triple helix, of the nature indicated above, had become well established by experiment by

1970. There is no time to comment on these studies*. However, one important fact relating to collagen which had not yet been given a proper theoretical basis stood out prominently — namely the role of hydroxyproline, an amino acid residue which occurs almost exclusively in collagen among all proteins. It should be mentioned, in this connection, that hydroxyproline (Hyp) is not one of the 20 amino acids that are coded by the genetic code, and it is not incorporated as such during the metabolic building up of the collagen chains in living tissues (see the review by Prockop (22) for further details). It is only incorporated as proline, and, after the formation of the triple helix, certain of the proline residues are hydroxylated by a special enzyme known as proline hydroxylase. It was also known that these Hyp residues occur only in the third position $\overset{Y}{/}$in the repeating sequence (— Gly — X — Y —) of the collagen structure, and that the hydroxylation is of the 4-_trans_ type universally in mammalian collagen, although other isomers such as 4-_cis_-Hyp and 3-Hyp, are observed in other systems. Therefore,

* Other articles in the two volumes mentioned in (1) and (2), as well as the extensive reviews by Traub and Piez (20), and Bornstein and Traub (21), may be referred to for further details.

the question occurs prominently — Why are such Hyp residues manufactured in the collagen structure, and what purpose do they serve in its physical chemistry or biochemistry? The problem was in the air in the field of collagen research for a long time; but an answer to this came from stereochemical considerations, as was found by us (23) in the early 1970's in Bangalore.

After I had moved to the Indian Institute of Science, Bangalore in 1971, we had a visit of Prof. R.S. Bhatnagar from USA, who had earlier worked on the biochemistry of the hydroxylation of proline with Prof. D.J. Prockop. During a discussion that took place over the table, Bhatnagar asked me pointedly as to what explanation I can give for the existence of hydroxyproline, and that too, with the particular 4-trans hydroxyl group that occurs in the side chain of Hyp residues. On inspecting the model, it became pretty clear to me that one of the water molecules, which was incorporated by Chandrasekaran and me into the collagen structure, had a free OH group, and that this was pointing towards the oxygen in the hydroxyl group of hydroxyproline, if the latter is present in

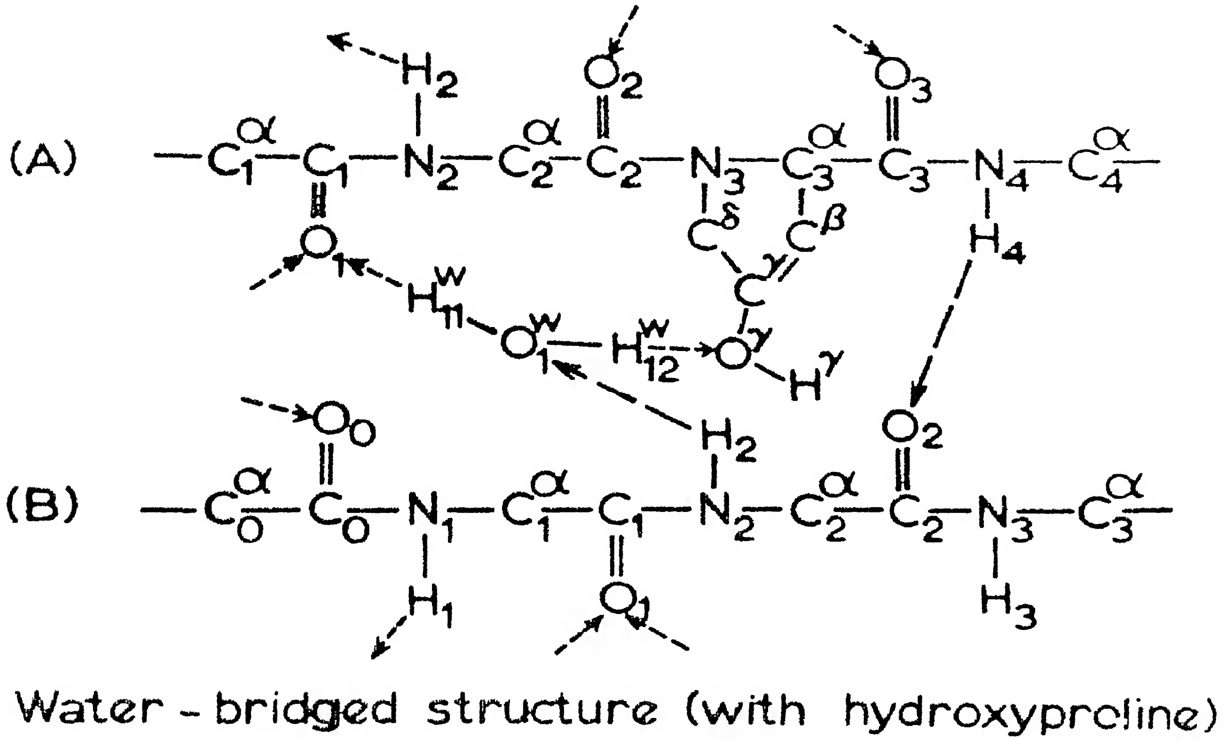


Fig. 12(a) : More hydrogen bond bridges introduced by hydroxyproline O H group. Note that in addition to those in Fig. 10(b), there is a linkage *via* hydrogen bonding between O_1^w and O^γ and that O^γ H^γ could serve for possible linkage with another triple helix. (From (3)).

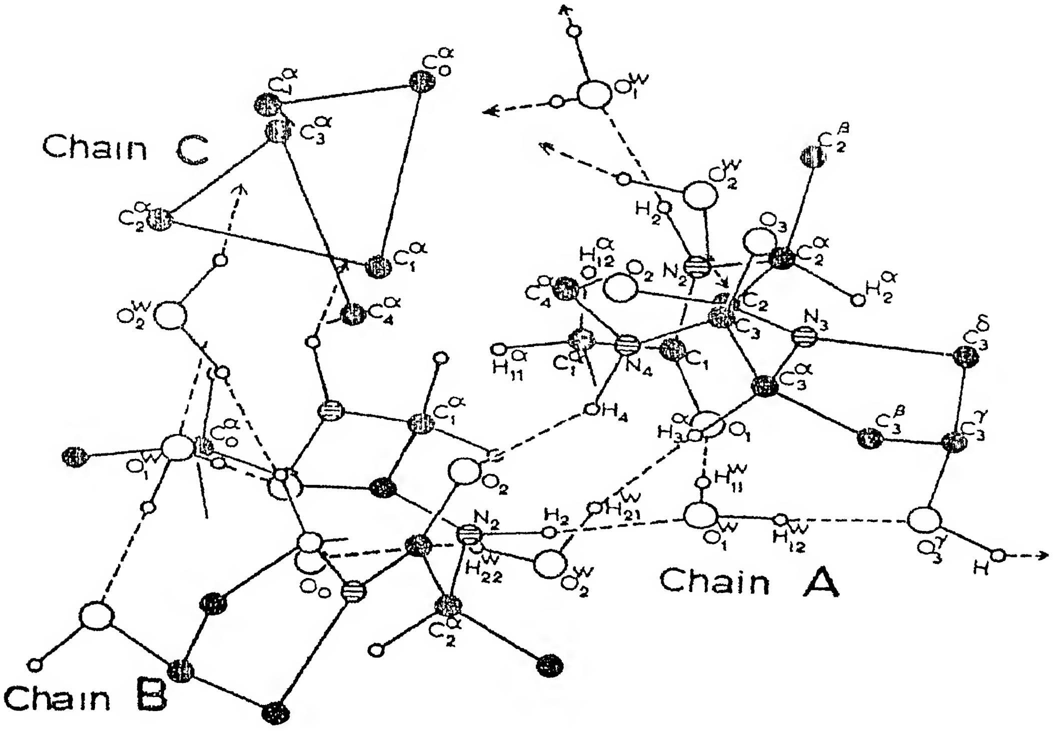


Fig.12(b): The above features are seen in projection in this figure. Note the occurrence of an extensive network of hydrogen bonds, linking neighbouring chains of the triple helical protofibril. (Reproduced from (3)).

the third position from glycine i.e., the position Y previous to the next glycine. Immediately, I asked Miss Manju Bansal, who had joined me as a research student, to take the coordinates of the structure as worked out by Chandrasekaran and see if it will fit in this extra hydrogen bond involving the hydroxyproline residue, if slight modifications are made in the orientation of the surrounding region in the molecule. It took Manju not more than two weeks to make a thorough examination of the possibilities, and to arrive at an acceptable set of coordinates for the collagen triple helix containing hydroxyproline in the third position Y, and/or proline in the second position X, in which this extra hydrogen bond is also made.

A schematic representation of all these hydrogen bonds, following the pattern in Figs. 10(a,b) is shown in Fig. 12(a, and an actual projection down the helical axis of the various groups involved in the hydrogen bonding network, making use of the water molecule and hydroxyproline OH group, is shown in Fig. 12(b). As mentioned before, there is a strong NH ... O hydrogen bond in the deep inside of the structure, and there is

a second network, consisting of NH ... O^w, and O^wH ... O involving the water oxygen O^w, linking the backbone. In addition, there is one more hydrogen bond linking the water oxygen O^w with O^γ of the Hyp side chain. Further, the hydroxyl group of the Hyp side chain is also pointed outwards from the centre of the triple helix, and could serve for interconnections between neighbouring triple helices via hydrogen bonds. Thus, hydroxyproline plays an important part not only in the stability of a single protofibril , but also in producing strong linkages between neighbouring protofibrils.

In this region of research also , experimental biochemists had been gathering information on the role of hydroxyproline. For instance, Berg and Prockop /(see 22) showed that the melting temperature of hydroxylated collagen was much higher than that of the unhydroxylated triple-helical protocollagen molecules. In fact, the melting temperature of the newly formed collagen molecule, without any Hyp residues in it, is as low as 25°C, while fully hydroxylated collagen has a melting point of 50°C. Since collagen is found mostly in

mammalian systems with body temperatures in the region of 30° to 40°C, these experiments clearly show that collagen can be chemically stable in the bodies of all animals only if it is hydroxylated. These experiments, and several others made using other techniques such as enzyme degradation etc., (see (22)) completely supported the results obtained from conformational theory that the real role of hydroxyproline is not chemical, but rather physical, in that its occurrence helps in the formation of a network of two, or three, hydrogen bonds for every three residues.

With this, the essentials of the stereochemistry of the secondary structure of the collagen molecule could be said to have been firmly established.

8. Primary sequence with glycine as every third residue

However, the complete resolution of the chemical sequence of the collagen molecule from the biochemical point of view became revealed only in the early 70's — thanks to the efforts of Karl Piez, who initiated project involving/a large international group with workers in Germany, Britain and USA, for/determining the primary sequence of the thousand-residue-long collagen

molecule, forming one of the three roughly equivalent chains in the triple helix (see the article by Piez (24) for a survey). When this primary structure was solved, it gave full support to the theoretical assumption, made in the very first structure, that Gly must occur at every third position, in the helical regions of the structure. It also indicated that Pro invariably occurs at the second position X (after Gly), and Hyp always occurs at the third position Y (before Gly), although Pro residues may remain partially unhydroxylated in the third position also.

Table 1 shows a portion of this primary sequence, in a number of different species of animals, all of which are seen to have this property. However, a more complete analysis of the protocollagen molecule indicated that collagen is primarily synthesized with an extension, both before the first residue having the regular sequence — Gly — X — Y — , and after the termination of this regular sequence. It is believed that these appendages help to align the three chains in proper justaposition, so as to initiate the formation of the triple helix.

With this, it can be said that all the mysteries of the collagen primary structure (amino acid sequence) and secondary structure (stereochemistry of the triple helix)

Table 1: Typical amino acid sequences* in a part of the helical region of the collagen peptide chain†

Residue	α1(I) rat	α1(I) calf	α1(I) chick	α1(II) steer	α2 rat	α2 calf
17	Gly	Gly	Gly	Gly	Gly	Gly
18	Pro	Pro	Pro	Val	Pro	Pro
19	Met	Met	Met	Met	Met	Met
20	Gly	Gly	Gly	Gly	Gly	Gly
21	Pro	Pro	Pro	Pro	Leu	Leu
22	Ser	Ser	Ala	Met	Met	Met
23	Gly	Gly	Gly	Gly	Gly	Gly
24	Pro	Pro	Pro	Pro	Pro	Pro
25	Arg	Arg	Arg	Arg	Arg	Arg
26	Gly	Gly	Gly	Gly	Gly	Gly
27	Leu	Leu	Leu	Pro	Pro	Pro
28	Hyp	Hyp	Hyp	Hyp	Hyp	Hyp
29	Gly	Gly	Gly	Gly	Gly	Gly
30	Pro	Pro	Pro	Pro	Ala	Ala
31	Hyp	Hyp	Hyp	Ala	Val	Ser
32	Gly	Gly	Gly	Gly	Gly	Gly
33	Ala	Ala	Ala	Ala	Ala	Ala
34	Hyp	Hyp	Hyp	Hyp	Hyp	Hyp
35	Gly	Gly	Gly	Gly	Gly	Gly
36	Pro	Pro	Pro	Pro	Pro	Pro
37	Gln	Gln	Gln	Gln	Gln	Gln
38	Gly	Gly	Gly	Gly	Gly	Gly
39	Phe	Phe	Phe	Phe	Phe	Phe
40	Gln	Gln	Gln	Gln	Gln	Gln
41	Gly	Gly	Gly	Gly	Gly	Gly
42	Pro	Pro	Pro	Asn	Pro	Pro
43	Hyp	Hyp	Hyp	Hyp	Ala	Hyp

*Note the occurrence of Gly as every third residue, of Pro always in the second position, and of Hyp in the third position, from Gly (The latter may be partially hydroxylated).

†Data from (24)

BSF Lecture
15.7.87

The Age of Computerization in Science

1. Introduction and preview

Prof. Narasimhaiah and friends,

I am deeply grateful to your President Prof. Narasimhaiah for having invited me to give the Valedictory Address of your Science Forum. I wondered what would be the best topic that I could choose and I decided that it should be something with which I am closely familiar and which is occupying my current interests. Therefore, I chose this subject, since computation has been the mainstream of our scientific activities during the last 40 years or more, and we could consider examples from our own fields of interest as illustrations for the lecture.

In this again, the scope of the talk will be restricted, since I am not a specialist in computer science as such. However, since I am quite familiar with its applications to mathematics, physics, biology, technology, and in recent years to logic, this would be a personal talk based on my experience with computation in these fields. I shall very briefly review the aspects of the growth of these subjects from the 40's to the 80's as seen by a scientist, who has used computer can now improvise games using computers which are even being utilized to train persons in the practice of games such as chess.

Perhaps it may be worthwhile for me to recall the changes that have taken place in the utilization of computers in scientific research. When I entered the research field in physics in 1942 under Prof. C.V. Raman, I was using only table calculators which were mechanical and which had to be cranked for each step of addition even for doing multiplication, and made sucn a grinding noise. In my doctorate study with Prof. Raman, I employed these almost incessantly for almost 1 - 2 years to grind out and test the formulae developed by us regarding the optical dispersion of crystals. Even when I went to Cambridge, England in 1949, the position was hardly different. Electrical calculators had come into use instead of mechanical ones, but these were only makes that imitated the mechanical calculators, and the only improvement was that the turning and shifting were done electrically by pressing a key, and they were much less noisy. Logarthmic tables still formed the essential assisting devices for doing calculations. However, two subjects of crystal structure determination and the study of biomolecular structure and reactivity, are inextricably mixed together now-a-days. In both these fields, we in India have played quite a useful part in the development of the subject and some thirty years ago the work produced in India was of the same level as that obtaining in the best laboratories. This was possible because advanced facilities and computation had not yet become the dominant part of such investigations. I shall illustrate how in both these fields continued activity at the top level has come out of our laboratories so that even today we are ready to take up work in the advancing front of knowledge.

Perhaps the most beautiful application of the consequences of Fourier transforms in crystallography is in the field of tomography, a subject that came into prominence in the 1970's, and in which I had the good fortune to write one of the earliest papers on the application of very specialised techniques based on Fourier transforms, for obtaining three-dimensional images from two-dimensional pictures. The theory which we developed, based of the developed world. But we lost the initiative mainly because computational facilities did not continue to be at the same level as the developed countries, and we are now slowly picking up the thread, thanks to the great stress being laid on thrust areas of research in recent years.

I shall conclude the lecture by giving you some ideas on computation as applied to logic, reasoning, and the processing of knowledge. The topic of artificial intelligence is much talked of now-a-days, but apparently, even in this intricate subject, newer approaches can be developed by trying to put in mathematical language, the essential principles of the reasoning process, considering it as a physical phenomenon subject to observation, analysis and systematization. I shall try to show you that the process of algorithmising "reasoning" can be done and that it can also be computerized. If there is time, we shall discuss what this means in the ultimate limit where we can ask the question, whether computers can replace men for producing original ideas. If I may forestall what our conclusion will be, I feel that, while the computers may increase speed, range, and scope, of thinking processes, it can never replace the originality and analytical power of the human mind which form the essential basis of intelligence.

along the length of the rope. In the case of a crystal, its x-ray diffraction pattern contains many reflections — of the order of 500 for a simple structure studied in the 50's, which increased to some 10,000 in the 60's, and 100,000 to 500,000 in the 70's and 80's. In principle, from the intensities of these reflections, it is possible to calculate the distribution of electron density in the repeating pattern of the crystals, and hence the positions of the atoms in the molecule, their spatial arrangement, the nature of the chemical bonds and so on. Fig.1 gives the essential principles involved in the mathematical reconstruction of the molecule from the x-ray diffraction pattern.

Fig.1 : Essential formulae of x-ray crystallography

In principle, the diffraction pattern, specified by $F(hk\ell)$, is the Fourier transform of the electron density distribution $\rho(xyz)$, as shown in Eq.(1), and this equation is reversible and can be used to give the latter, in terms of the former, as in Eq.(3). But in practice, there is an unknown factor $\alpha(hk\ell)$ in (2), which has to be determined by special techniques, or has to be guessed by the crystallographer and then refined and improved by calculation. In the 1950's, there were no hard and fast methods for determining the phases in a general case, but computers were being applied for the arithmetical manipulations required in Eqs (3a) and (3b), and the increase in power of the computers has gone side by side with the increase in size of the molecular structures that were solved by x-ray crystallography. In the 60's, mathematical theories for the determination of α from $|I(hk\ell)|$ were developed, and the phase problem became more amenable to computerization. So also, the process of mounting a crystal and rotating it to various positions and recording reflection intensities, were also computerized and automatic diffractometers came into being. Thus, the labour involved in a crystal structure determination has been minimized to an almost unbelievable extent during the last 30 years, by the progressive use of computers at every level of experiment and theory. In fact, people are talking of a completely computerized structure determination in which the crystal is just fixed to the x-ray apparatus and everything else is computer-implemented and the outcome is a series of diagrams indicating positions of the atoms in the crystal.

Growth of x-ray crystallography in size and complexity

Years	Nature of molecules (examples)	Number of atoms in a molecule		Number of reflections N (Maximum)	Computers	Memory
		Maximum	In India			
1950's	Amino acids, peptides vitamins	50	20	1,000	Home-made Elliott-803	8K
1960's	Vitamin B12 Proteins – first examples	1,000	50	10,000	IBM-7090 CDC-3600	64K
1970's	Enzymes Immunoglobulin Transfer RNA	10,000	100	100,000	IBM-370 DEC-10	512K
1980's	Viruses 50 S Ribosome particle	100,000	10,000	500,000	CRAY-1 CRAY-2	Unlimited

I have been talking of molecules and their sizes. Fig.4 gives an idea of the relative sizes of molecules, and

Fig.4 : Molecular size comparison .

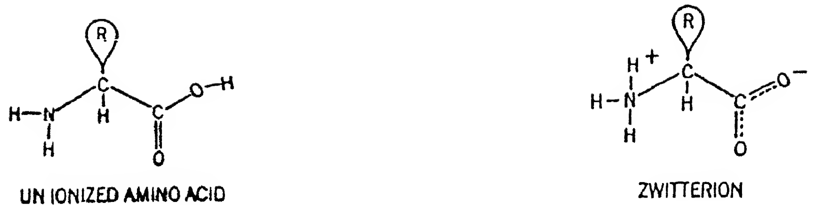
the diagram is self-explanatory. The molecules shown at the top are simple organic compounds — for example, lysine is an amino acid, and heme is a planar compound of great importance in the blood protein hemoglobin which is shown below. In the latter, the thick rod-like structures shown, actually consist of a chain of atoms composed of a sequence of amino acid residues and they are folded together in this convoluted form to make the biologically active blood protein hemoglobin, which transports oxygen from the lungs to the tissues. The location of the heme molecule is also shown by the black discs in the figure. The increase in size of the protein molecule, in relation to small organic molecules, is clear from the figure. This jump in size took place in the early 60's and this has gone up by another factor of 10-50 during the last twenty years, and even "living molecules" like viruses have been tackled in the last few years.

3. Biomolecular structures

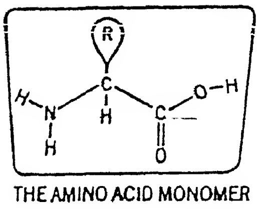
Leaving aside crystallography as such, we may ask the question — "Why does the molecular chain in large molecules, such as hemoglobin, take such a convoluted structure, and how can this be understood from chemistry and physics, and how can this be correlated with biological activity?" Since this talk is not purely about biochemistry or biophysics, I shall only make brief comments about this. The essence of the process of protein folding is contained in Fig.5. As will be seen from this, a protein molecule

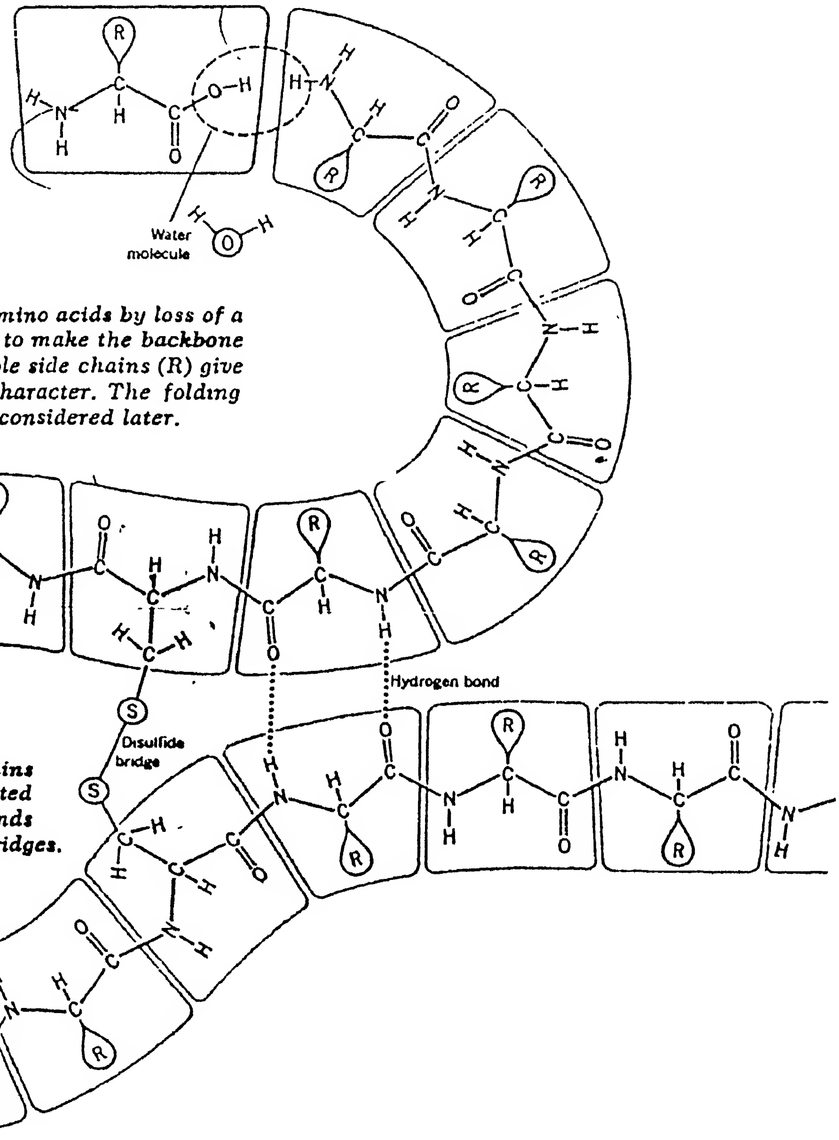
Fig.5 : Folding of polypeptide chains

is a long chain of repeating peptide units, each of which is one of 20 different types, but which are closely similar, and differ only in what are known as side groups (R). The chain is flexible, and therefore it folds back on itself and gets stabilized by "chemical bonds" such as —S—S— linkages, or "physical bonds" such as electrostatic interactions and hydrogen bonds, occurring between different parts of the chain, which hold them together in the form of a loop or similar structures. It is natural that the whole molecule will take up a compact globular configuration having nearly



The amino acids exist in solution as doubly charged zwitterions





The peptide bond formed between amino acids by loss of a water molecule joins identical units to make the backbone of the polypeptide chain. The variable side chains (R) give each protein chain its distinctive character. The folding of the chain in three dimensions is considered later.

Polypeptide chains are cross-connected by hydrogen bonds and disulfide bridges.

the two strands of the double helix, or the three strands of the triple helix, as the case may be. Fig. 7(b) shows the twist of the three strands around one another in the collagen triple helix. This twist is regular because the peptide sequence in collagen is based on repeating tripeptide (—Gly—X—Y—) and half of X and Y are also based on the same amino acid residue proline. On the other hand, the sequence is irregular and different in myoglobin, and this favours the formation of single α-helical segments, which are then folded up to form a globular structure as in Fig. 7(a).

Fig.7 (a) Peptide chain configuration in myoglobin which is closely related to hemoglobin.

(b) Twisted rope configuration of the collagen triple helix.

Obviously, in checking the validity of such structures, and of the wide range of possibilities that may occur, computers have played a very important part. However, the early studies in the 50's were mostly done without computers, and in fact a pioneering study of the principles governing these, which were published from Madras in 1963, was completely done with mechanical desk calculators, which took almost a year to complete (see Fig.8). Now-a-days, these will take a fraction of a second on the best computers. I shall not mention much about these theoretical ideas except to indicate that the two principal considerations that were put forward by us at that time have played a dominant part in all techniques involving molecular packing, and the fitting of molecular chains to electron density diagrams obtained from crystal structure determination, ever since. These are the use of dihedral angles (rotation angles), such as (ϕ, ψ) of Fig. 8(a), and the idea that non-bonded atoms cannot come closer than certain specified limiting distances, which was used for preparing the diagram in Fig. 8(b). These have been so well

Fig.8 (a) Rotation angles ϕ and ψ specifying the configuration of a pair of peptide units.

(b) The (ϕ, ψ)-plot of the observed configurations in myoglobin (marked as small circles.)

computerized that, now-a-days one can just feed in electron density and the chemical nature of the molecule and ask the computer to shift, turn and twist the molecule until the best fit with the electron density is obtained, subject to the restrictions imposed by the limiting distances mentioned above. More about this later.

4. Crystallography and Biology with special reference to drug design

The precise techniques that have been developed for energy minimization, and for the working out of the best possible configurations of molecules, have resulted in a body of knowledge which has vastly expanded our knowledge of the physical chemistry, molecular biophysics and biochemistry, of the biological systems. I shall illustrate these by a few examples.

We had seen that the heme molecule in hemoglobin is located in between the different protein chains and the knowledge obtained by crystallography about the particular amino acid residues near the location of the heme had led to a clear understanding of the mechanism of the function and reactivity of the protein for oxygen transport in blood. Simultaneously, it has also become clear that many biological conditions associated with the malfunction of hemoglobin can be attributed to changes in one or two amino acid residues produced by mutation. Fig.9 shows the location of the mutated residues for a large number of

Fig.9 Mutation sites in pathalogical hemoglobins.

pathalogical hemoglobins. It will be seen that, though the amino sequence in a linear peptide chain may not show it, the x-ray crystallographic three-dimensional structure shows that they often occur close to the region where the heme is held, or in the gaps between neighbouring molecules. Apart from this general feature, the knowledge obtained from crystallography has profoundly affected biochemical investigations for the understanding of such diseases of the blood, and has led even to the design of suitable chemical drugs for alleviating the symptoms.

Turning to more fundamental problems in biology, such as the action of enzymes, the chemists had formulated the theory that enzymes, which regulate all biological reactions, did so by forming a close molecular fit with the substrate molecule concerned. This was a conjuncture in the past, although a very reasonable one and much substantiated by all available knowledge. However, a confirming evidence in this matter has come only from cfystallographic studies. One of the earliest proteins to be solved, namely lysozyme which is shown in Fig. 10, displays this feature remarkably well.

Fig. 10 : Space-filling molecular models of free lysozyme (left) and when bound to its substrate (right).

The first picture is that of the protein which is seen to have a cleft on its left, and the second picture is of the enzyme-substrate complex. It will be seen that the substrate and enzyme form a perfect fit and the theoretical postulation of molecular fit has been demonstrated in all detail by three-dimensional x-ray crystallography.

I will therefore restrict myself to the subject of molecular fit, and give a few examples where crystallography, in combination with conformational theorey, has been applied to obtain information regarding this. Fig.11 is a study made by Prof. Viswamitra of the Indian Institute of Science in which he determined the molecular structure of an antibiotic TANDEM, shown in the top half of Fig. 11, which acts by binding with DNA. He could then show that this fits very well to the DNA double helix, as shown in the lower half. This idea has been employed in a number of studies for finding the basis of drug-molecule interactions. Such crystallographic studies

Fig.11 : Space-filling models of (a) the antibiotic TANDEM and (b) its complex with DNA.

have even been extended to virus molecules and the interaction of antivirul agents with the virus. Fig. 12 is an example of a very recent study made by Prof. Rossman in Purdue University, USA.

Fig. 12 : Crystallographically determined mode of binding of antiviral agent WIN to virus protein VP1.

Not only does it show that the viral protein has a cavity which can be fitted by the antiviral agent WIN, so that the normal biochemical reactions involved in the reproduction of the virus molecule are inhibited, but it is also an example of the extent to which crystallography has progressed. The virus itself required 200,000 reflections for its structure determination. After that, the mode of attachment of the drug was determined by using what is known as a difference-Fourier synthesis, in which the difference in intensities between x-ray diffraction patterns, of the pure virus and the virus-drug complex, was used to find out the location of the drug molecule within the virus. Fig. 13 shows the electron density distribution of the drug molecule as reconstructed by a computer. It will be seen

Fig. 13. Front page of Science magazine containing an illustration of the molecule of an antiviral agent fitted to the computer-displayed electron density cage.

that it effectively indicates a cage within which the molecule must be situated, and the exact location of the molecule and its conformation can also be varied and fitted in, by employing the computer for on-line operation. I had the good fortune to advise some of the workers in this field of computer-fitting of molecules in USA in 1977-78, and this approach is universally employed both in crystallography and conformational theory now-a-days, and "computer graphics terminals" are even commercially available for this purpose.

5. Conformational theory for molecular fit

Coming back to theoretical calculations, Fig. 14 is an example where the crystal structure determination of an antibiotic (daunomycin) was the starting point of an attempt to fit the molecule to the DNA double helix. The work was done in Prof. Pullman's laboratory in Paris, and as the figure will show, the fitting can be done readily by opening out the double helix in the region where the drug makes its entry. I wish to emphasize that this has not been done by constructing

Fig. 14: Daunomycin fitted to DNA by energy minimization on a computer

a model, and then turning and twisting the molecular structure so as to obtain the fit, but purely via the use of a computer. On the other hand, the fitting of an inhibitor, β-P-P-P, to the active site of the enzyme thermolysin, which was carried out by Prof. Rao in Bangalore, was done, not automatically on a computer, but using the computer without on-line interaction. The investigator starts with a structure close to a reasonable fit, and then modifies it step by step, by feeding in and taking out the data from the computer in the form of drawings. This study showed that the inhibitor enters the enzyme in one configuration and then both its conformation, and that of the enzyme in the local region, are modified so as to obtain the best fit. The reason why I am showing this picture is to indicate that some of the latest ideas of this type have been computerized right in our country and interesting consequences have been deduced. However, even with a computer at one's disposal, this is a long process requiring weeks or months. If only an interactive graphic display system, as has been used in all the advanced

Fig.15: Mode of attachment of the inhibitor β-P-P-P which imitates the substrate of the enzyme thermolysin.

countries, were made available, the time could be reduced to probably a week or two. In fact, as already mentioned, such systems are available commercially in USA and Japan, and I believe that it is time that such up-to-date equipment be made available to the capable scientists of our country.

Emergence of the new field QSAR

Such studies on molecular fit have opened many new avenues in the field of rational drug design, because drugs invariably either promote, or inhibit, particular biological reactions, either in the organism producing infection, or in the host for producing resistance. In fact, a new field entitled Quantitative Structure Activity Relationship (QSAR) has emerged during the last few years, and strategies are being developed for actively pursuing this for drug design. Fig. 16 shows the title page of a very interesting paper, on protein crystallography and computer graphics, published in the German journal Angewandte Chemie (Applied Chemistry), which emphasizes this aspect of

Fig. 16: Title page and flow chart about QSAR reproduced from a recent article.

biomolecular physics. As will be seen from one of the flow charts in this report, computer graphics and theoretical calculations play as important a part as the determination of three-dimensional structure by x-ray crystallography. Even regular symposia are held on this subject of QSAR, as will be seen from Fig. 16.

6. Computerized Tomography

Turning to the subject of tomography, my interests in this field dates from 1970 when some very interesting work was done along with Dr. Lakshminarayanan, at the University of Chicago on the application of Fourier transforms and the novel technique of convolutions for this purpose. Fig. 17 shows the title page of our first paper on the subject entitled "Reconstruction of substance from shadow"

Fig. 17: Fourier transform equations formulated in polar coordinates for axial tomography.

As you will see from this, the Fourier transform Eq.(1), which is basic to this subject, is the same as that used in x-ray crystallography. However, although crystallographers had applied this technique for finding out the three-dimensional structure of virus particles from two-dimensional electron microscopic images in different directions, we suggested that this reconstruction is much faster by employing polar coordinates for the Fourier transform as in Eqs. (2a,b), using the principle of axial tomography. In these, $g(\ell\ ;\ \theta)$ is the linear shadowgraphs at θ, and $f(r, \phi)$ is the two-dimensional image that is reconstructed for a two-dimensional section at right angles to the axis of rotation.

This was followed by another paper published in the Proceedings of the National Academy of Sciences, USA, whose title page is shown in Fig. 18. In this, the way in which a series of two-dimensional images can be converted into a three-dimensional reconstruction by axial tomography is shown in the diagram. In addition, a new modification of the Fourier method, which we called as the "convolution method" was presented in this paper, and its basic mathematics is shown in Eqs. (3), (4) and (5).

Fig. 18: Principle of the convolution method of reconstruction in axial tomography.

In this, the reconstructed two-dimensional image $f(r\ ,\ \phi)$ is a simple integral of the density functions $g'(\ell\ ;\ \theta)$ at θ, which represent the modified shadowgraphs at θ, producea as a result of the convolution process shown in Eqs (4a,b). The detailed way in which the modified function is calculated is by the procedure given in Eqs (5a,b). The mathematics is not quite relevant to this lecture, but what is important is that, as a consequence, the reconstruction can be performed almost a hundred times faster than the Fourier method and, as will be seen from Fig. 19, it is also about ten times more accurate.

Fig. 19: Tables 1 and 3 reproduced from the PNAS paper showing the distinct advantage in speed and accuracy of the convolution method.

Tables 1 and 3 of Fig. 19 are the comparative data of the accuracy and computing time required for the different methods of reconstruction , namely Cartesian Fourier transform (FTC), Polar Fourier transform (FTP) and Convolution (CON). The most interesting feature is that the larger the amount of data that is available from measurements for reconstruction, the faster is the convolution method relative to the Fourier method. This technique is applied widely in CAT-scan instruments particularly for medical radiography.

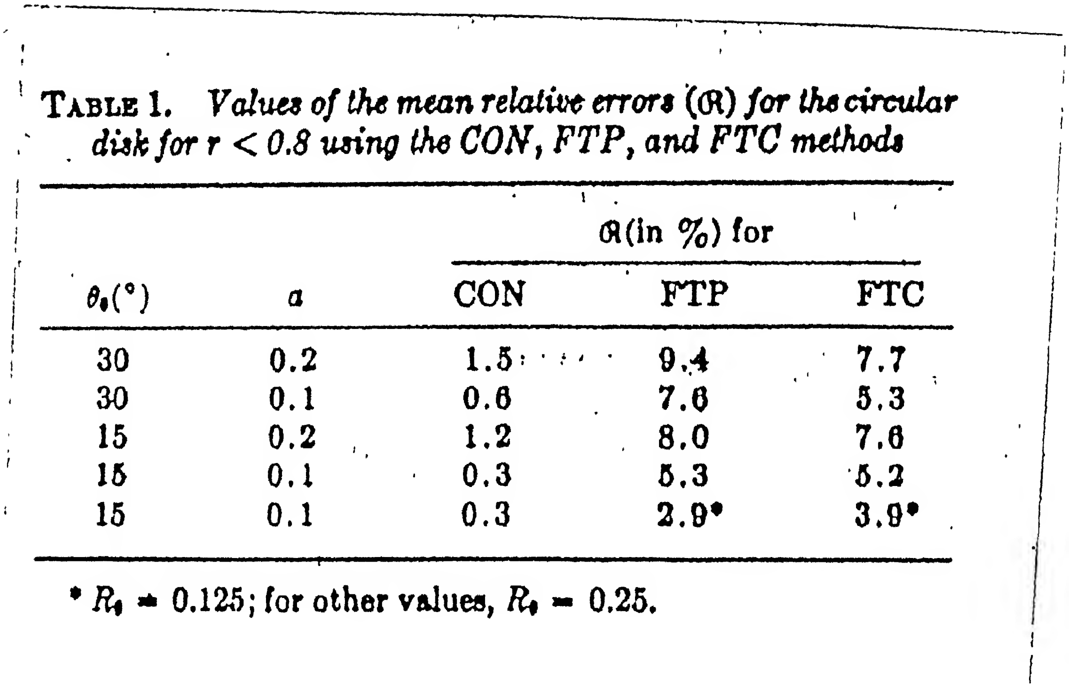


TABLE 1. *Values of the mean relative errors ($\mathcal{R}$) for the circular disk for $r < 0.8$ using the CON, FTP, and FTC methods*

		$\mathcal{R}$(in %) for		
θ_0(°)	a	CON	FTP	FTC
30	0.2	1.5	9.4	7.7
30	0.1	0.6	7.6	5.3
15	0.2	1.2	8.0	7.6
15	0.1	0.3	5.3	5.2
15	0.1	0.3	2.9*	3.9*

* R_0 = 0.125; for other values, R_0 = 0.25.

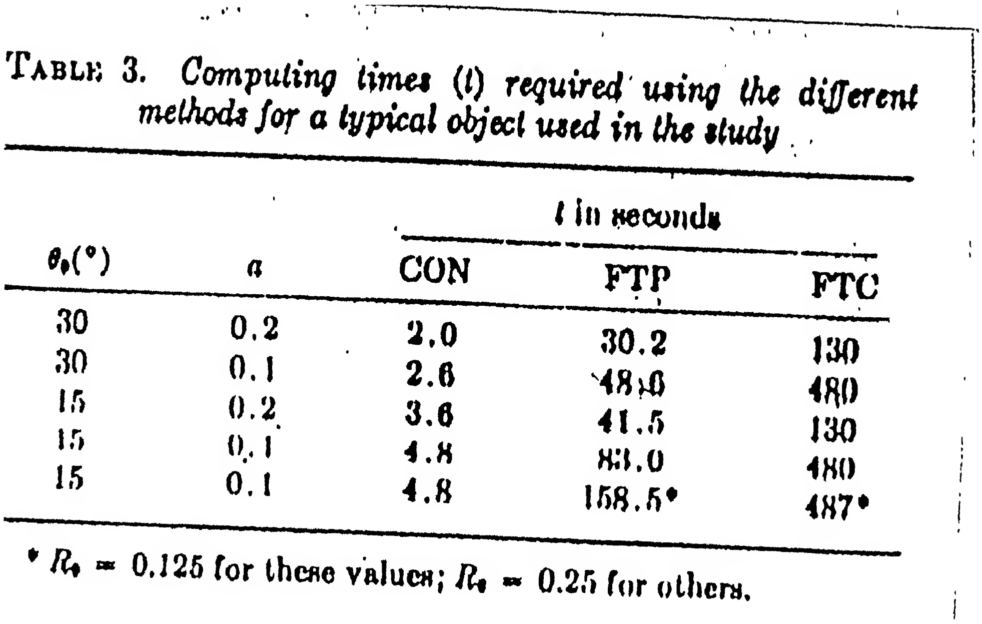
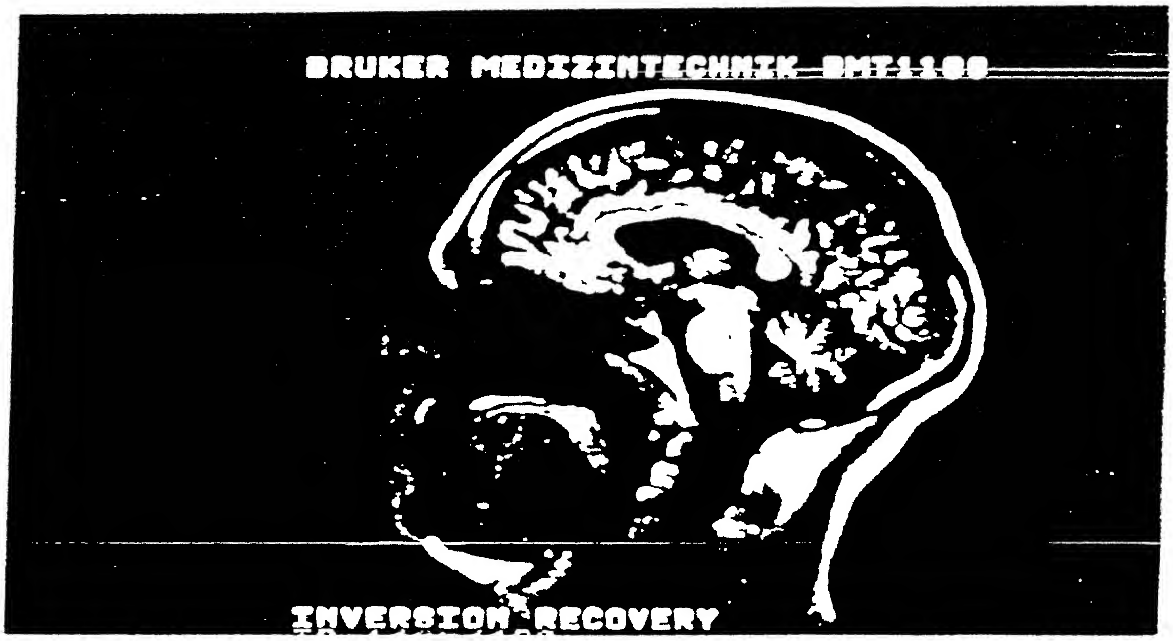
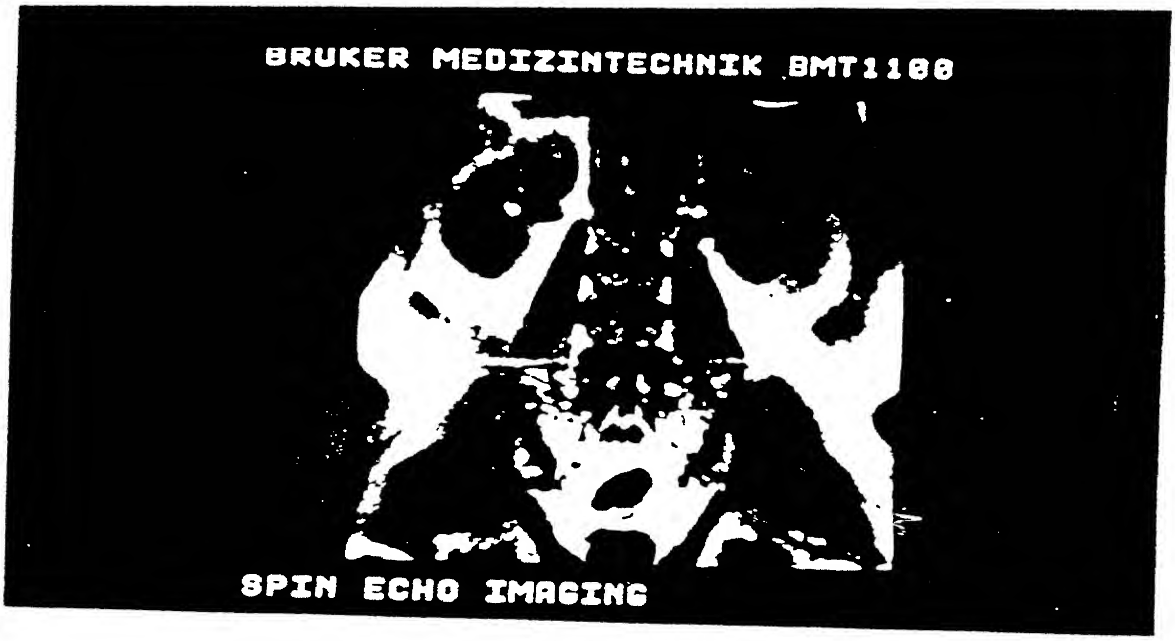


TABLE 3. *Computing times (t) required using the different methods for a typical object used in the study*

		t in seconds		
θ_0(°)	a	CON	FTP	FTC
30	0.2	2.0	30.2	130
30	0.1	2.6	48.6	480
15	0.2	3.6	41.5	130
15	0.1	4.8	83.0	480
15	0.1	4.8	158.5*	487*

* R_0 = 0.125 for these values; R_0 = 0.25 for others.


BRUKER MEDIZINTECHNIK BMT1100
INVERSION RECOVERY


BRUKER MEDIZINTECHNIK BMT1100
SPIN ECHO IMAGING

from these two physical methods, other imaging techniques have come up, such as positron emission tomography, which are also finding increasing application, particularly in the early detection of cancer. We will not spend more time on this subject except to say that this field has been one, in which, not only were new mathematical approaches very effective, but the computational aspects were the ones that decided the practical applicability of the technique. In fact, a complete scientific journal entitled "The Journal of Computed Tomography" is being published, having come up to the tenth volume in 1986, which shows the outstandingly significant role of computation in modern medical research and practice. I believe that the convolution method has still greater potentialities by the inclusion of what is known as parallel processing in computation, which is being rapidly developed in recent years.

7. Computerization of Logic

We shall now discuss the last topic of my talk, namely computers for logic. This is a topic in which I have been working for the last ten years and much of it is still in the process of development. However, I thought I should include

but which at the same time is directly interpretable on the computer via Boolean algebra. I hope I will have time to indicate at least in a broad way its nature and its advantage.

Essentially, all computers have a central processing unit (CPU) in which the fundamental arithmetical operations of addition and multiplication are carried out, but not all instructions that have to be given to computers, for automatic control of machines and so on, are arithmetical, or in the form in which they can be converted into ordinary algebra. For instance, it will be necessary to compare one thing with another, and if the two are equal, it will carry out one operation while if the two are different, another operation is to be performed. This is the simplest of what may be called a "logical operation". So also, we may be given a list of related entities — such as a is related to b, c and d, b is related to c, e and f, c is related to a, d and f, and so on. Then we have to answer the question — which are the entities that are related by the given relation to b, c and d ? In standard computer practice, these can be worked out, but the procedures depend upon what are termed "goal-seeking searches",

BOOLEAN ALGEBRA

$a \oplus b = c$

a \ b	1	0
1	1	1
0	1	0

$a \otimes b = c$

a \ b	1	0
1	1	0
0	0	0

$a^c = b$

a	b
1	0
0	1

Boolean algebra is closed with respect to sum, product and complement.

LOGICAL TRUTH VALUES

a OR b = c

a \ b	T	F
T	T	T
F	T	F

a AND b =c

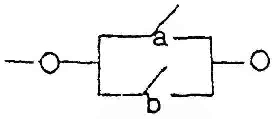
a \ b	T	F
T	T	F
F	F	F

NOT a = b

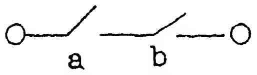
a	b
T	F
F	T

There are only two truth values T, F, in classical logic.

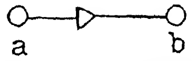
SWITCHING CIRCUITS



OR GATE



AND GATE



Inverter(NOT)

a \ b	ON	OFF
ON	ON	ON
OFF	ON	OFF

a \ b	ON	OFF
ON	ON	OFF
OFF	OFF	OFF

a	b
ON	OFF
OFF	ON

The principles of these switching circuits are the basis of all computers.

The three sets of tables are isomorphic to one another.

gives rise to the operation of subtraction by reversing the equation for it, as in (1), and similarly, multiplication leads to division by reversal of Eq.(2); as given in Fig. 24, both of which are included in standard arithmetic. It is supposed that these "reverse" operations have no analogs in Boolean algebra which has only the properties of a "Boolean ring" in the parlance of modern algebra. However, this is not the case. On can, in fact, reverse the operations of Boolean sum

Fig. 24: BA-2 algebra is essentially needed for the representation of reverse relations in classical logic.

and Boolean product, by using tne corresponding analogs of the above two equations (1) and (2) for subtraction and division, and thus obtain interesting consequences. These are shown in Fig. 24, and it will be seen that both types of reversal are possible for Boolean arithmetic, but that they lead to two more new "truth values", D and X, in addition to T and F of classical logic.

Thus, BA-1 algebra is insufficient for representing fully all consequences in classical logic, in that T and F are not

enough to represent the extended algebra of the logical truth tables, and the reversal of the operations OR and AND leads to four different "truth values". Without going into details, I may say that it is possible to build a matrix algebra, in which the four states T, F, D, X are represented by 2-element Boolean vectors as shown in Eq.(3), which we have named BA-2. The use of "or" and "and" between T and F for the states D and X is justified by the descriptions given in Eq.(3). There is no time to give details.

This is not the end of the story, but the beginning of a whole new approach to logic. Perhaps the most significant feature of this approach is the recognition and symbolization of such reverse operators, which lead to what we have called "unary" relations. Thus, the truth tables for OR and AND. can be taken over as 2x2 matrices, as in Eqs.(1a) and (2a) of Fig.25, and the answer to a query such as, "If a AND b is given to be true, and a is given as T (or F), what is b in each case?" can be written in the language of Boolean vector-matrix formalism (BVMF) as in Eqs.(1b,c) and (2b,c). As will be seen from these, all the essential conclusions

Fig. 25: Unary relations in BA-2 algebra and in the general theory of relations.

made in Fig. 24 about the consequences of the reversed operators are reproduced in the examples (1b,c) and (2b,c) by this formalism.

There is no time to talk about various applications of this vector-matrix formalism, using 2-vectors and 3-vectors, for the branches of logic known as propositional calculus and quantifier calculus. I shall, however, indicate the great convenience of the vector-matrix formalism in what is known as predicate calculus, which is most relevant to practical applications of logic. The basic formula that connects two sets of objects can be represented in this formalism by a matrix R_{ij} as shown in Eq.(4). Thus, suppose A_1, A_2 ... A_m are a number of persons and B_1, B_2 ... B_n are the possible cities from which they come. Then, if A_1 belongs to the city B_2, we say that the element R_{12} of the relation "coming from city"is 1, and 0 otherwise. Thus, we obtain an array of R_{ij} ($i = 1$ to m, $j = 1$ to n), each of which is a Boolean number, having only the values 1, 0. Then, if $m = 5$, and we have a subset of A's consisting of A_1, A_3, A_5, then we can represent them by a 5-element Boolean vector $(a_1, a_2, a_3, a_4, a_5) = (1\ 0\ 1\ 0\ 1)$.

BSF Lecture
21.7.87

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}]{\text{Professor of}} \underset{\sim}{p}$$

$$GM1 = \begin{array}{c|cccc} & p_1 & p_2 & p_3 & p_4 \\ \hline s_1 & 1 & 0 & 1 & 0 \\ s_2 & 0 & 1 & 1 & 0 \\ s_3 & 1 & 0 & 0 & 1 \\ s_4 & 0 & 0 & 0 & 1 \\ s_5 & 1 & 1 & 1 & 0 \end{array}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}]{\text{Student of}} \underset{\sim}{s}$$

$$GM3 = \\ = GT(GM1) \quad \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}^c]{\text{Not professor of}} \underset{\sim}{p}$$

$$GM2 = \\ = GC(GM1) \quad \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}^c]{\text{Not student of}} \underset{\sim}{s}$$

$$GM4 = \\ = GTC(GM1) \quad \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

1. Who are all the professors who teach the students s_1 and s_2 ?

s_1, s_2 gives GVS = (1 1 0 0 0) — Vector representing given students

GUNPDT(GVS,GM1) = GVP — Relation "student to professor" GM1 applied (1a)

GVP = (1 1 1 0) — Ans: p_1, p_2, p_3 — Vector representing required professors. (1b)

2. Which professors among p_2 and p_4 teach at least one of the students s_1, s_3 ?

Input GVP1 = (0 1 0 1), GVS1 = (1 0 1 0 0)

GUNPDT(GVS1,GM1) = GVP, GVP = (1 0 1 1) : p_1, p_3, p_4 teach either s_1 or s_2 (2a)

GVIDYA(GVP,GVP1) = GVP2 , GVP2 = (0 0 0 1) : Of these, only p_4 is contained in the input set p_2, p_4 (2b)

we give some examples in Fig. 27.

The statements there are fairly self-explanatory in view of what has been said earlier. In these problems also, we deal with a set of five students and four professors, but consider two relations that can exist between them, namely $\underset{\sim}{P}$ = "professor of", and $\underset{\sim}{E}$ = "examiner of", whose matrices are given in (1), as GMP and GME. (Note that this GMP is different from that in Fig. 26.) Now we ask two questions as given in (a) and (b) of Fig. 27. It requires some effort to differentiate between the two in common language, but in BVMF, the two are given by quite different formulae, as in (3) and (4) of Fig. 27. Thereafter, we can leave it to the computer, and interestingly, the output vectors GVPA and GVPB, for the questions in (a) and (b), are quite different as given in (5). These mean that the answer to the first problem is that there is only one professor p_4 who satisfies both the conditions of this problem, while the answer to the second problem is that three professors p_1, p_2, p_4 satisfy the given conditions. I am giving this example to indicate how even such tricky problems can be formulated in almost a single line in BVMF, and can be computerized via MATLOG.

Fig. 27: Examples of logical "tongue-twisters" solved using BVMF and MATLOG

As already mentioned in the beginning, this does not at all mean that computers can replace human thought. The computer can only work out the answer to a query regarding the validity of a conclusion in the forms : "yes, "no", "does not follow from the assumption" or "contradictory to known facts"; but it cannot suggest what questions should be asked to clinch the issue, if the deduction, is found to be inconclusive. Much can be said on this inter-relationship between semantics and logic, but, in essence, we can say that the latter, which is the grammar of reasoning, can be computerized, while the former continues to be still in the realm of human endeavour.

(see Fig.8). Now-a-days, these will take a fraction of a second on the best computers. I shall not mention much about these theoretical ideas except to indicate that the two principal considerations that were put forward by us at that time have played a dominant part in all techniques involving molecular packing, and the fitting of molecular chains to electron density diagrams obtained from crystal structure determination, ever since. These are the use of dihedral angles (rotation angles), such as (ϕ, ψ) of Fig. 8(a), and the idea that non-bonded atoms cannot come closer than certain specified limiting distances, which was used for preparing the diagram in Fig. 8(b). These have been so well

Fig.8 (a) Rotation angles ϕ and ψ specifying the configuration of a pair of peptide units.

(b) The (ϕ, ψ)-plot of the observed configurations in myoglobin (marked as small circles.)

computerized that, now-a-days one can just feed in electron density and the chemical nature of the molecule and ask the computer to shift, turn and twist the molecule until the best fit with the electron density is obtained, subject to the restrictions imposed by the limiting distances mentioned above. More about this later.

4. Crystallography and Biology with special reference to drug design

The precise techniques that have been developed for energy minimization, and for the working out of the best possible configurations of molecules, have resulted in a body of knowledge which has vastly expanded our knowledge of the physical chemistry, molecular biophysics and biochemistry, of the biological systems. I shall illustrate these by a few examples.

We had seen that the heme molecule in hemoglobin is located in between the different protein chains and the knowledge obtained by crystallography about the particular amino acid residues near the location of the heme had led to a clear understanding of the mechanism of the function and reactivity of the protein for oxygen transport in blood. Simultaneously, it has also become clear that many biological conditions associated with the malfunction of hemoglobin can be attributed to changes in one or two amino acid residues produced by mutation. Fig.9 shows the location of the mutated residues for a large number of pathalogical hemoglobins. It will be seen that, though the amino

Fig.9 Mutation sites in pathalogical hemoglobins.

sequence in a linear peptide chain may not show it, the x-ray crystallographic three-dimensional structure shows that they often occur close to the region where the heme is held, or in the gaps between neighbouring molecules. Apart from this general feature, the knowledge obtained from crystallography has profoundly affected biochemical investigations for the understanding of such diseases of the blood, and has led even to the design of suitable chemical drugs for alleviating the symptoms.

Turning to more fundamental problems in biology, such as the action of enzymes, the chemists had formulated the theory that enzymes, which regulate all biological reactions, did so by forming a close molecular fit with the substrate molecule concerned. This was a conjuncture in the past, although a very reasonable one and much substantiated by all available knowledge. However, a confirming evidence in this matter has come only from cfystallographic studies. One of the earliest proteins to be solved, namely lysozyme which is shown in Fig. 10, displays this feature remarkably well.

Fig. 10 : Space-filling molecular models of free lysozyme (left) and when bound to its substrate (right).

The first picture is that of the protein which is seen to have a cleft on its left, and the second picture is of the enzyme-substrate complex. It will be seen that the substrate and enzyme form a perfect fit and the theoretical postulation of molecular fit has been demonstrated in all detail by three-dimensional x-ray crystallography.

I will therefore restrict myself to the subject of molecular fit, and give a few examples where crystallography, in combination with conformational theorey, has been applied to obtain information regarding this. Fig.11 is a study made by Prof. Viswamitra of the Indian Institute of Science in which he determined the molecular structure of an antibiotic TANDEM, shown in the top half of Fig. 11, which acts by binding with DNA. He could then show that this fits very well to the DNA double helix, as shown in the lower half. This idea has been employed in a number of studies for finding the basis of drug-molecule interactions. Such crystallographic studies

Fig.11 : Space-filling models of (a) the antibiotic TANDEM and (b) its complex with DNA.

have even been extended to virus molecules and the interaction of antivirul agents with the virus. Fig. 12 is an example of a very recent study made by Prof. Rossman in Purdue University, USA.

Fig. 12 : Crystallographically determined mode of binding of antiviral agent WIN to virus protein VP1.

Not only does it show that the viral protein has a cavity which can be fitted by the antiviral agent WIN, so that the normal biochemical reactions involved in the reproduction of the virus molecule are inhibited, but it is also an example of the extent to which crystallography has progressed. The virus itself required 200,000 reflections for its structure determination. After that, the mode of attachment of the drug was determined by using what is known as a difference-Fourier synthesis, in which the difference in intensities between x-ray diffraction patterns, of the pure virus and the virus-drug complex, was used to find out the location of the drug molecule within the virus. Fig. 13 shows the electron density distribution of the drug molecule as reconstructed by a computer. It will be seen that it effectively indicates a cage within which the molecule

Fig. 13. Front page of Science magazine containing an illustration of the molecule of an antiviral agent fitted to the computer-displayed electron density cage.

must be situated, and the exact location of the molecule and its conformation can also be varied and fitted in, by employing the computer for on-line operation. I had the good fortune to advise some of the workers in this field of computer-fitting of molecules in USA in 1977-78, and this approach is universally employed both in crystallography and conformational theory now-a-days, and "computer graphics terminals" are even commercially available for this purpose.

5. Conformational theory for molecular fit

Coming back to theoretical calculations, Fig. 14 is an example where the crystal structure determination of an antibiotic (daunomycin) was the starting point of an attempt to fit the molecule to the DNA double helix. The work was done in Prof. Pullman's laboratory in Paris, and as the figure will show, the fitting can be done readily by opening out the double helix in the region where the drug makes its entry. I wish to emphasize that this has not been done by constructing

Fig. 14: Daunomycin fitted to DNA by energy minimization on a computer

a model, and then turning and twisting the molecular structure so as to obtain the fit, but purely <u>via</u> the use of a computer. On the other hand, the fitting of an inhibitor, β-P-P-P, to the active site of the enzyme thermolysin, which was carried out by Prof. Rao in Bangalore, was done, not automatically on a computer, but using the computer without on-line interaction. The investigator starts with a structure close to a reasonable fit, and then modifies it step by step, by feeding in and taking out the data from the computer in the form of drawings. This study showed that the inhibitor enters the enzyme in one configuration and then both its conformation, and that of the enzyme in the local region, are modified so as to obtain the best fit. The reason why I am showing this picture is to indicate that some of the latest ideas of this type have been computerized right in our country and interesting consequences have been deduced. However, even with a computer at one's disposal, this is a long process requiring weeks or months. If only an interactive graphic display system, as has been used in all the advanced

Fig.15: Mode of attachment of the inhibitor β-P-P-P which imitates the substrate of the enzyme thermolysin.

countries, were made available, the time could be reduced to probably a week or two. In fact, as already mentioned, such systems are available commercially in USA and Japan, and I believe tnat it is time that such up-to-date equipment be made available to the capable scientists of our country.

Emergence of the new field QSAR

Such studies on molecular fit have opened many new avenues in the field of rational drug design, because drugs invariably either promote, or inhibit, particular biological reactions, either in the organism producing infection, or in the host for producing resistance. In fact, a new field entitled Quantitative Structure Activity Relationship (QSAR) has emerged during the last few years, and strategies are being developed for actively pursuing this for drug design. Fig. 16 shows the title page of a very interesting paper, on protein crystallography and computer graphics, published in the German journal Angewandte

Fig. 16: Title page and flow chart about QSAR reproduced from a recent article.

Chemie (Applied Chemistry), which emphasizes this aspect of biomolecular physics. As will be seen from one of the flow charts in this report, computer graphics and theoretical calculations play as important a part as the determination of three-dimensional structure by x-ray crystallography. Even regular symposia are held on this subject of QSAR, as will be seen from Fig. 16.

6. Computerized Tomography

Turning to the subject of tomography, my interests in this field dates from 1970 when some very interesting work was done along with Dr. Lakshminarayanan, at the University of Chicago on the application of Fourier transforms and the novel technique of convolutions for this purpose. Fig. 17 shows the title page of our first paper on the subject entitled "Reconstruction of substance from shadow"

Fig. 17: Fourier transform equations formulated in polar coordinates for axial tomography.

As you will see from this, the Fourier transform Eq.(1), which is basic to this subject, is the same as that used in x-ray crystallography. However, although crystallographers had applied this technique for finding out the three-dimensional structure of virus particles from two-dimensional electron microscopic images in different directions, we suggested that this reconstruction is much faster by employing polar coordinates for the Fourier transform as in Eqs. (2a,b), using the principle of axial tomography. In these, $g(\ell ; \theta)$ is the linear shadowgraphs at θ, and $f(r, \phi)$ is the two-dimensional image that is reconstructed for a two-dimensional section at right angles to the axis of rotation.

This was followed by another paper published in the Proceedings of the National Academy of Sciences, USA, whose title page is shown in Fig. 18. In this, the way in which a series of two-dimensional images can be converted into a three-dimensional reconstruction by axial tomography is shown in the diagram. In addition, a new modification of the Fourier method, which we called as the "convolution method" was presented in this paper, and its basic mathematics is shown in Eqs. (3), (4) and (5).

Fig. 18: Principle of the convolution method of reconstruction in axial tomography.

In this, the reconstructed two-dimensional image $f(r , \phi)$ is a simple integral of the density functions $g'(\ell ; \theta)$ at θ, which represent the modified shadowgraphs at Θ , produced as a result of the convolution process shown in Eqs (4a,b). The detailed way in which the modified function is calculated is by the procedure given in Eqs (5a,b). The mathematics is not quite relevant to this lecture, but what is important is that, as a consequence, the reconstruction can be performed almost a hundred times faster than the Fourier method and, as will be seen from Fig. 19, it is also about ten times more accurate.

Fig. 19: Tables 1 and 3 reproduced from the PNAS paper showing the distinct advantage in speed and accuracy of the convolution method.

Tables 1 and 3 of Fig. 19 are the comparative data of the accuracy and computing time required for the different methods of reconstruction , namely Cartesian Fourier transform (FTC), Polar Fourier transform (FTP) and Convolution (CON). The most interesting feature is that the larger the amount of data that is available from measurements for reconstruction, the faster is the convolution method relative to the Fourier method. This technique is applied widely in CAT-scan instruments particularly for medical radiography.

TABLE 1. *Values of the mean relative errors* ($\mathcal{R}$) *for the circular disk for $r < 0.8$ using the CON, FTP, and FTC methods*

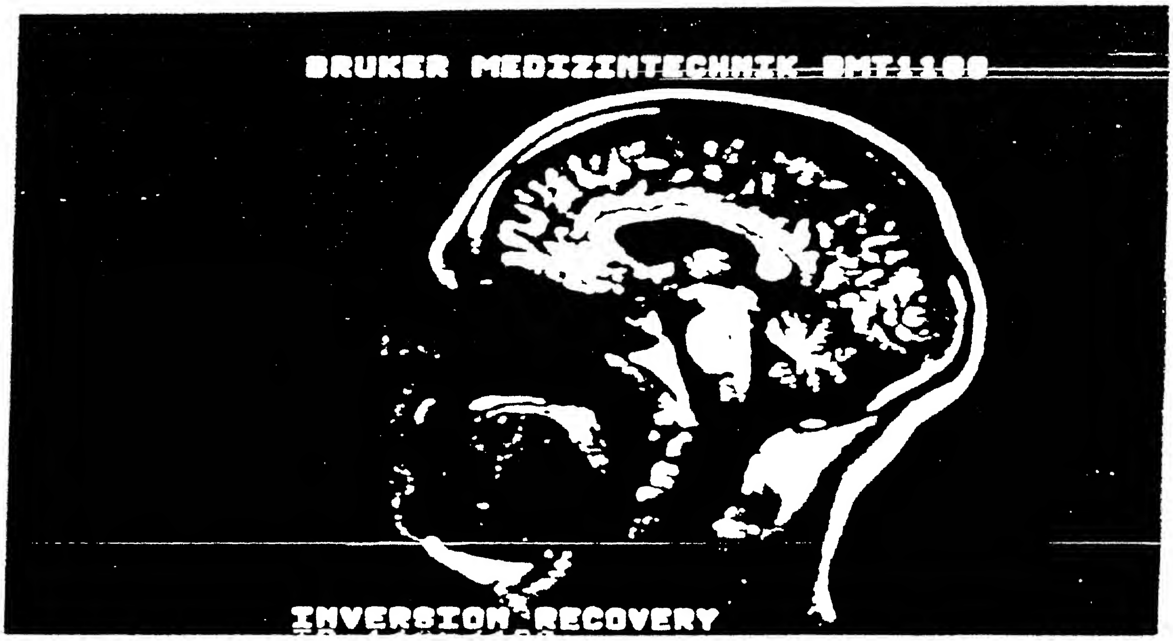
θ_s(°)	a	$\mathcal{R}$(in %) for CON	FTP	FTC
30	0.2	1.5	9.4	7.7
30	0.1	0.6	7.6	5.3
15	0.2	1.2	8.0	7.6
15	0.1	0.3	5.3	5.2
15	0.1	0.3	2.9*	3.9*

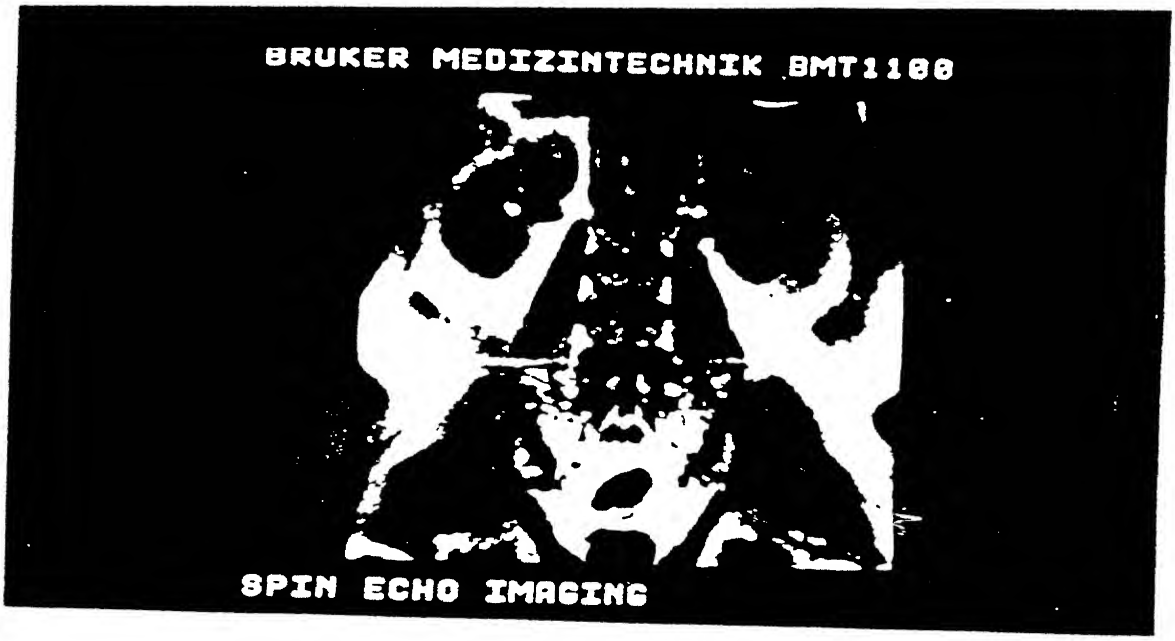
* R_s = 0.125; for other values, R_s = 0.25.

TABLE 3. *Computing times* (t) *required using the different methods for a typical object used in the study*

θ_s(°)	a	t in seconds CON	FTP	FTC
30	0.2	2.0	30.2	130
30	0.1	2.6	48.6	480
15	0.2	3.6	41.5	130
15	0.1	4.8	83.0	480
15	0.1	4.8	158.5*	487*

* R_s = 0.125 for these values; R_s = 0.25 for others.


BRUKER MEDIZINTECHNIK BMT1100
INVERSION RECOVERY


BRUKER MEDIZINTECHNIK BMT1100
SPIN ECHO IMAGING

from these two physical methods, other imaging techniques have come up, such as positron emission tomography, which are also finding increasing application, particularly in the early detection of cancer. We will not spend more time on this subject except to say that this field has been one, in which, not only were new mathematical approaches very effective, but the computational aspects were the ones that decided the practical applicability of the technique. In fact, a complete scientific journal entitled "The Journal of Computed Tomography" is being published, having come up to the tenth volume in 1986, which shows the outstandingly significant role of computation in modern medical research and practice. I believe that the convolution method has still greater potentialities by the inclusion of what is known as parallel processing in computation, which is being rapidly developed in recent years.

7. Computerization of Logic

We shall now discuss the last topic of my talk, namely computers for logic. This is a topic in which I have been working for the last ten years and much of it is still in the process of development. However, I thought I should include

but which at the same time is directly interpretable on the computer via Boolean algebra. I hope I will have time to indicate at least in a broad way its nature and its advantage.

Essentially, all computers have a central processing unit (CPU) in which the fundamental arithmetical operations of addition and multiplication are carried out, but not all instructions that have to be given to computers, for automatic control of machines and so on, are arithmetical, or in the form in which they can be converted into ordinary algebra. For instance, it will be necessary to compare one thing with another, and if the two are equal, it will carry out one operation while if the two are different, another operation is to be performed. This is the simplest of what may be called a "logical operation". So also, we may be given a list of related entities — such as a is related to b, c and d, b is related to c, e and f, c is related to a, d and f, and so on. Then we have to answer the question — which are the entities that are related by the given relation to b, c and d ? In standard computer practice, these can be worked out, but the procedures depend upon what are termed "goal-seeking searches",

BOOLEAN ALGEBRA

$a \oplus b = c$

a \ b	1	0
1	1	1
0	1	0

$a \otimes b = c$

a \ b	1	0
1	1	0
0	0	0

$a^c = b$

a	b
1	0
0	1

Boolean algebra is closed with respect to sum, product and complement.

LOGICAL TRUTH VALUES

a OR b = c

a \ b	T	F
T	T	T
F	T	F

a AND b =c

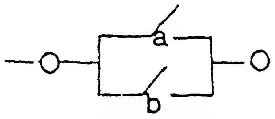
a \ b	T	F
T	T	F
F	F	F

NOT a = b

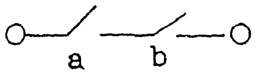
a	b
T	F
F	T

There are only two truth values T, F, in classical logic.

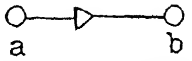
SWITCHING CIRCUITS



OR GATE



AND GATE



Inverter(NOT)

a \ b	ON	OFF
ON	ON	ON
OFF	ON	OFF

a \ b	ON	OFF
ON	ON	OFF
OFF	OFF	OFF

a	b
ON	OFF
OFF	ON

The principles of these switching circuits are the basis of all computers.

The three sets of tables are isomorphic to one another.

gives rise to the operation of subtraction by reversing the equation for it, as in (1), and similarly, multiplication leads to division by reversal of Eq.(2), as given in Fig. 24, both of which are included in standard arithmetic. It is supposed that these "reverse" operations have no analogs in Boolean algebra which has only the properties of a "Boolean ring" in the parlance of modern algebra. However, this is not the case. On can, in fact, reverse the operations of Boolean sum and Boolean product, by using tne corresponding analogs of the above two equations (1) and (2) for subtraction and division, and thus obtain interesting consequences. These are shown in Fig. 24, and it will be seen that both types of reversal are possible for Boolean arithmetic, but that they lead to two more new "truth values", D and X, in addition to T and F of classical logic.

Fig. 24: BA-2 algebra is essentially needed for the representation of reverse relations in classical logic.

Thus, BA-1 algebra is insufficient for representing fully all consequences in classical logic, in that T and F are not

enough to represent the extended algebra of the logical truth tables, and the reversal of the operations OR and AND leads to four different "truth values". Without going into details, I may say that it is possible to build a matrix algebra, in which the four states T, F, D, X are represented by 2-element Boolean vectors as shown in Eq.(3), which we have named BA-2. The use of "or" and "and" between T and F for the states D and X is justified by the descriptions given in Eq.(3). There is no time to give details.

This is not the end of the story, but the beginning of a whole new approach to logic. Perhaps the most significant feature of this approach is the recognition and symbolization of such reverse operators, which lead to what we have called "unary" relations. Thus, the truth tables for OR and AND. can be taken over as 2x2 matrices, as in Eqs.(1a) and (2a) of Fig.25, and the answer to a query such as, "If a AND b is given to be true, and a is given as T (or F), what is b in each case?" can be written in the language of Boolean vector-matrix formalism (BVMF) as in Eqs.(1b,c) and (2b,c). As will be seen from these, all the essential conclusions

Fig. 25: Unary relations in BA-2 algebra and in the general theory of relations.

made in Fig. 24 about the consequences of the reversed operators are reproduced in the examples (1b,c) and (2b,c) by this formalism.

There is no time to talk about various applications of this vector-matrix formalism, using 2-vectors and 3-vectors, for the branches of logic known as propositional calculus and quantifier calculus. I shall, however, indicate the great convenience of the vector-matrix formalism in what is known as predicate calculus, which is most relevant to practical applications of logic. The basic formula that connects two sets of objects can be represented in this formalism by a matrix R_{ij} as shown in Eq.(4). Thus, suppose A_1, A_2 ... A_m are a number of persons and B_1, B_2 ... B_n are the possible cities from which they come. Then, if A_1 belongs to the city B_2, we say that the element R_{12} of the relation "coming from city"is 1, and 0 otherwise. Thus, we obtain an array of R_{ij} ($i = 1$ to m, $j = 1$ to n), each of which is a Boolean number, having only the values 1, 0. Then, if $m = 5$, and we have a subset of A's consisting of A_1, A_3, A_5, then we can represent them by a 5-element Boolean vector $(a_1, a_2, a_3, a_4, a_5) = (1\ 0\ 1\ 0\ 1)$.

BSF Lecture
21.7.87

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}]{\text{Professor of}} \underset{\sim}{p}$$

$$\text{GM1} = \begin{array}{c|cccc} & p_1 & p_2 & p_3 & p_4 \\ \hline s_1 & 1 & 0 & 1 & 0 \\ s_2 & 0 & 1 & 1 & 0 \\ s_3 & 1 & 0 & 0 & 1 \\ s_4 & 0 & 0 & 0 & 1 \\ s_5 & 1 & 1 & 1 & 0 \end{array}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}]{\text{Student of}} \underset{\sim}{s}$$

$$\text{GM3} = \text{GT(GM1)} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}^c]{\text{Not professor of}} \underset{\sim}{p}$$

$$\text{GM2} = \text{GC(GM1)} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}^c]{\text{Not student of}} \underset{\sim}{s}$$

$$\text{GM4} = \text{GTC(GM1)} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

1. Who are all the professors who teach the students s_1 and s_2 ?

s_1, s_2 gives GVS = (1 1 0 0 0) — Vector representing given students

GUNPDT(GVS,GM1) = GVP — Relation "student to professor" GM1 applied (1a)

GVP = (1 1 1 0) — Ans: p_1, p_2, p_3 — Vector representing required professors. (1b)

2. Which professors among p_2 and p_4 teach at least one of the students s_1, s_3 ?

Input GVP1 = (0 1 0 1), GVS1 = (1 0 1 0 0)

GUNPDT(GVS1,GM1) = GVP, GVP = (1 0 1 1) : p_1, p_3, p_4 teach either s_1 or s_2 (2a)

GVIDYA(GVP,GVP1) = GVP2 , GVP2 = (0 0 0 1) : Of these, only p_4 is contained in the input set p_2, p_4 (2b)

we give some examples in Fig. 27.

The statements there are fairly self-explanatory in view of what has been said earlier. In these problems also, we deal with a set of five students and four professors, but consider two relations that can exist between them, namely $\underset{\sim}{P}$ = "professor of", and $\underset{\sim}{E}$ = "examiner of", whose matrices are given in (1), as GMP and GME. (Note that this GMP is different from that in Fig. 26.) Now we ask two questions as given in (a) and (b) of Fig. 27. It requires some effort to differentiate between the two in common language, but in BVMF, the two are given by quite different formulae, as in (3) and (4) of Fig. 27. Thereafter, we can leave it to the computer, and interestingly, the output vectors GVPA and GVPB, for the questions in (a) and (b), are quite different as given in (5). These mean that the answer to the first problem is that there is only one professor p_4 who satisfies both the conditions of this problem, while the answer to the second problem is that three professors p_1, p_2, p_4 satisfy the given conditions. I am giving this example to indicate how even such tricky problems can be formulated in almost a single line in BVMF, and can be computerized via MATLOG.

Fig. 27: Examples of logical "tongue-twisters" solved using BVMF and MATLOG

As already mentioned in the beginning, this does not at all mean that computers can replace human thought. The computer can only work out the answer to a query regarding the validity of a conclusion in the forms : "yes, "no", "does not follow from the assumption" or "contradictory to known facts"; but it cannot suggest what questions should be asked to clinch the issue, if the deduction is found to be inconclusive. Much can be said on this inter-relationship between semantics and logic, but, in essence, we can say that the latter, which is the grammar of reasoning, can be computerized, while the former continues to be still in the realm of human endeavour.

TABLE 1. *Values of the mean relative errors ($\mathcal{R}$) for the circular disk for $r < 0.8$ using the CON, FTP, and FTC methods*

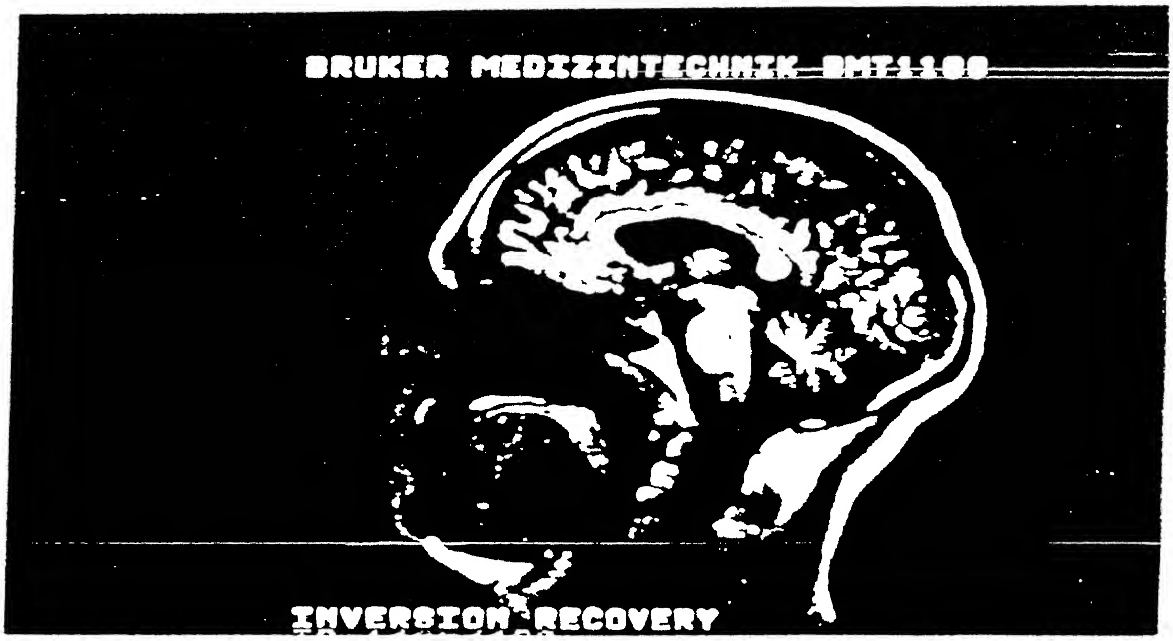
		$\mathcal{R}$(in %) for		
θ_0(°)	a	CON	FTP	FTC
30	0.2	1.5	9.4	7.7
30	0.1	0.6	7.6	5.3
15	0.2	1.2	8.0	7.6
15	0.1	0.3	5.3	5.2
15	0.1	0.3	2.9*	3.9*

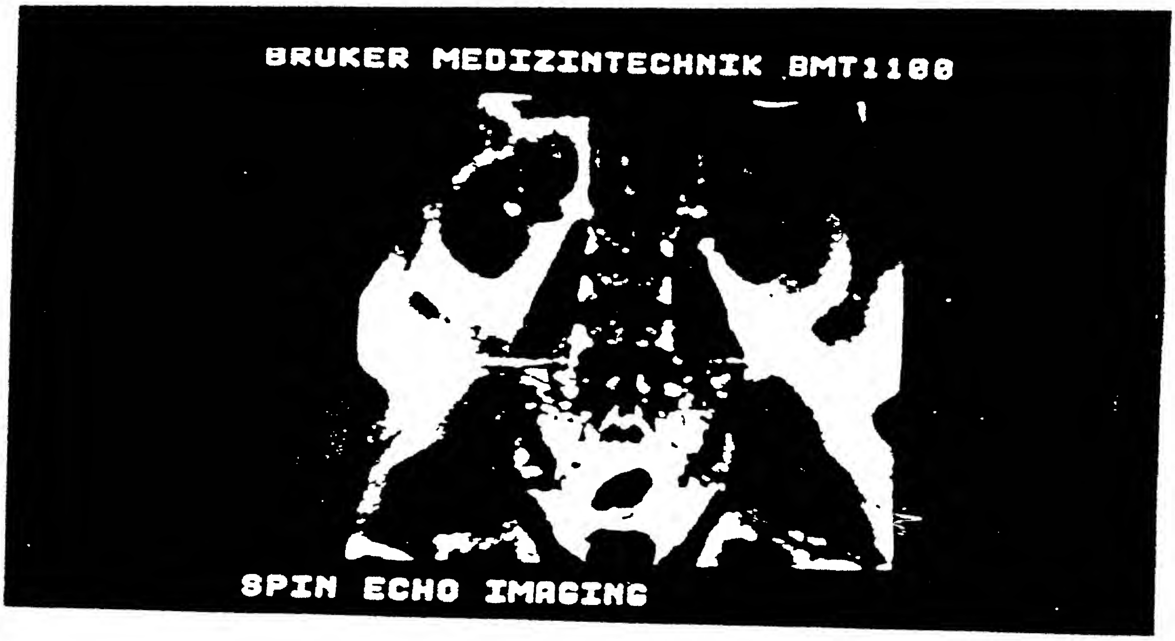
* R_0 = 0.125; for other values, R_0 = 0.25.

TABLE 3. *Computing times (t) required using the different methods for a typical object used in the study*

		t in seconds		
θ_0(°)	a	CON	FTP	FTC
30	0.2	2.0	30.2	130
30	0.1	2.6	48.6	480
15	0.2	3.6	41.5	130
15	0.1	4.8	83.0	480
15	0.1	4.8	158.5*	487*

* R_0 = 0.125 for these values; R_0 = 0.25 for others.


BRUKER MEDIZINTECHNIK BMT1100
INVERSION RECOVERY


BRUKER MEDIZINTECHNIK BMT1100
SPIN ECHO IMAGING

from these two physical methods, other imaging techniques have come up, such as positron emission tomography, which are also finding increasing application, particularly in the early detection of cancer. We will not spend more time on this subject except to say that this field has been one, in which, not only were new mathematical approaches very effective, but the computational aspects were the ones that decided the practical applicability of the technique. In fact, a complete scientific journal entitled "The Journal of Computed Tomography" is being published, having come up to the tenth volume in 1986, which shows the outstandingly significant role of computation in modern medical research and practice. I believe that the convolution method has still greater potentialities by the inclusion of what is known as parallel processing in computation, which is being rapidly developed in recent years.

7. Computerization of Logic

We shall now discuss the last topic of my talk, namely computers for logic. This is a topic in which I have been working for the last ten years and much of it is still in the process of development. However, I thought I should include

but which at the same time is directly interpretable on the computer via Boolean algebra. I hope I will have time to indicate at least in a broad way its nature and its advantage.

Essentially, all computers have a central processing unit (CPU) in which the fundamental arithmetical operations of addition and multiplication are carried out, but not all instructions that have to be given to computers, for automatic control of machines and so on, are arithmetical, or in the form in which they can be converted into ordinary algebra. For instance, it will be necessary to compare one thing with another, and if the two are equal, it will carry out one operation while if the two are different, another operation is to be performed. This is the simplest of what may be called a "logical operation". So also, we may be given a list of related entities/such as a is related to b, c and d, b is related to c, e and f, c is related to a, d and f, and so on. Then we have to answer the question – which are the entities that are related by the given relation to b, c and d ? In standard computer practice, these can be worked out, but the procedures depend upon what are termed "goal-seeking searches",

BOOLEAN ALGEBRA

$a \oplus b = c$

a \ b	1	0
1	1	1
0	1	0

$a \otimes b = c$

a \ b	1	0
1	1	0
0	0	0

$a^c = b$

a	b
1	0
0	1

Boolean algebra is closed with respect to sum, product and complement.

LOGICAL TRUTH VALUES

a OR b = c

a \ b	T	F
T	T	T
F	T	F

a AND b =c

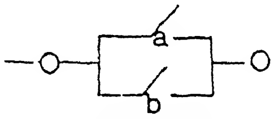
a \ b	T	F
T	T	F
F	F	F

NOT a = b

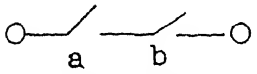
a	b
T	F
F	T

There are only two truth values T, F, in classical logic.

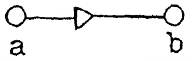
SWITCHING CIRCUITS



OR GATE



AND GATE



Inverter(NOT)

a \ b	ON	OFF
ON	ON	ON
OFF	ON	OFF

a \ b	ON	OFF
ON	ON	OFF
OFF	OFF	OFF

a	b
ON	OFF
OFF	ON

The principles of these switching circuits are the basis of all computers.

The three sets of tables are isomorphic to one another.

gives rise to the operation of subtraction by reversing the equation for it, as in (1), and similarly, multiplication leads to division by reversal of Eq.(2), as given in Fig. 24, both of which are included in standard arithmetic. It is supposed that these "reverse" operations have no analogs in Boolean algebra which has only the properties of a "Boolean ring" in the parlance of modern algebra. However, this is not the case. On can, in fact, reverse the operations of Boolean sum

Fig. 24: BA-2 algebra is essentially needed for the representation of reverse relations in classical logic.

and Boolean product, by using tne corresponding analogs of the above two equations (1) and (2) for subtraction and division, and thus obtain interesting consequences. These are shown in Fig. 24, and it will be seen that both types of reversal are possible for Boolean arithmetic, but that they lead to two more new "truth values", D and X, in addition to T and F of classical logic.

Thus, BA-1 algebra is insufficient for representing fully all consequences in classical logic, in that T and F are not

enough to represent the extended algebra of the logical truth tables, and the reversal of the operations OR and AND leads to four different "truth values". Without going into details, I may say that it is possible to build a matrix algebra, in which the four states T, F, D, X are represented by 2-element Boolean vectors as shown in Eq.(3), which we have named BA-2. The use of "or" and "and" between T and F for the states D and X is justified by the descriptions given in Eq.(3). There is no time to give details.

This is not the end of the story, but the beginning of a whole new approach to logic. Perhaps the most significant feature of this approach is the recognition and symbolization of such reverse operators, which lead to what we have called "unary" relations. Thus, the truth tables for OR and AND can be taken over as 2x2 matrices, as in Eqs.(1a) and (2a) of Fig.25, and the answer to a query such as, "If a AND b is given to be true, and a is given as T (or F), what is b in each case?" can be written in the language of Boolean vector-matrix formalism (BVMF) as in Eqs.(1b,c) and (2b,c). As will be seen from these, all the essential conclusions

Fig. 25: Unary relations in BA-2 algebra and in the general theory of relations.

made in Fig. 24 about the consequences of the reversed operators are reproduced in the examples (1b,c) and (2b,c) by this formalism.

There is no time to talk about various applications of this vector-matrix formalism, using 2-vectors and 3-vectors, for the branches of logic known as propositional calculus and quantifier calculus. I shall, however, indicate the great convenience of the vector-matrix formalism in what is known as predicate calculus, which is most relevant to practical applications of logic. The basic formula that connects two sets of objects can be represented in this formalism by a matrix R_{ij} as shown in Eq.(4). Thus, suppose A_1, A_2 ... A_m are a number of persons and B_1, B_2 ... B_n are the possible cities from which they come. Then, if A_1 belongs to the city B_2, we say that the element R_{12} of the relation "coming from city" is 1, and 0 otherwise. Thus, we obtain an array of R_{ij} ($i = 1$ to m, $j = 1$ to n), each of which is a Boolean number, having only the values 1, 0. Then, if $m = 5$, and we have a subset of A's consisting of A_1, A_3, A_5, then we can represent them by a 5-element Boolean vector $(a_1, a_2, a_3, a_4, a_5) = (1\ 0\ 1\ 0\ 1)$.

BSF Lecture
21.7.87

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}]{\text{Professor of}} \underset{\sim}{p}$$

$$GM1 = \begin{array}{c|cccc} & p_1 & p_2 & p_3 & p_4 \\ \hline s_1 & 1 & 0 & 1 & 0 \\ s_2 & 0 & 1 & 1 & 0 \\ s_3 & 1 & 0 & 0 & 1 \\ s_4 & 0 & 0 & 0 & 1 \\ s_5 & 1 & 1 & 1 & 0 \end{array}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}]{\text{Student of}} \underset{\sim}{s}$$

$$GM3 = \\ = GT(GM1) \quad \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}^c]{\text{Not professor of}} \underset{\sim}{p}$$

$$GM2 = \\ = GC(GM1) \quad \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}^c]{\text{Not student of}} \underset{\sim}{s}$$

$$GM4 = \\ = GTC(GM1) \quad \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

1. Who are all the professors who teach the students s_1 and s_2 ?

s_1, s_2 gives GVS = (1 1 0 0 0) — Vector representing given students

GUNPDT(GVS,GM1) = GVP — Relation "student to professor" GM1 applied (1a)

GVP = (1 1 1 0) — Ans: p_1, p_2, p_3 — Vector representing required professors. (1b)

2. Which professors among p_2 and p_4 teach at least one of the students s_1, s_3 ?

Input GVP1 = (0 1 0 1), GVS1 = (1 0 1 0 0)

GUNPDT(GVS1,GM1) = GVP, GVP = (1 0 1 1) : p_1, p_3, p_4 teach either s_1 or s_2 (2a)

GVIDYA(GVP,GVP1) = GVP2 , GVP2 = (0 0 0 1) : Of these, only p_4 is contained in the input set p_2, p_4 (2b)

we give some examples in Fig. 27.

The statements there are fairly self-explanatory in view of what has been said earlier. In these problems also, we deal with a set of five students and four professors, but consider two relations that can exist between them, namely $\underset{\sim}{P}$ = "professor of", and $\underset{\sim}{E}$ = "examiner of", whose matrices are given in (1), as GMP and GME. (Note that this GMP is different from that in Fig. 26.) Now we ask two questions as given in (a) and (b) of Fig. 27. It requires some effort to differentiate between the two in common language, but in BVMF, the two are given by quite different formulae, as in (3) and (4) of Fig. 27. Thereafter, we can leave it to the computer, and interestingly, the output vectors GVPA and GVPB, for the questions in (a) and (b), are quite different as given in (5). These mean that the answer to the first problem is that there is only one professor p_4 who satisfies both the conditions of this problem, while the answer to the second problem is that three professors p_1, p_2, p_4 satisfy the given conditions. I am giving this example to indicate how even such tricky problems can be formulated in almost a single line in BVMF, and can be computerized via MATLOG.

Fig. 27: Examples of logical "tongue-twisters" solved using BVMF and MATLOG

As already mentioned in the beginning, this does not at all mean that computers can replace human thought. The computer can only work out the answer to a query regarding the validity of a conclusion in the forms : "yes, "no", "does not follow from the assumption" or "contradictory to known facts"; but it cannot suggest what questions should be asked to clinch the issue, if the deduction is found to be inconclusive. Much can be said on this inter-relationship between semantics and logic, but, in essence, we can say that the latter, which is the grammar of reasoning, can be computerized, while the former continues to be still in the realm of human endeavour.

enough to represent the extended algebra of the logical truth tables, and the reversal of the operations OR and AND leads to four different "truth values". Without going into details, I may say that it is possible to build a matrix algebra, in which the four states T, F, D, X are represented by 2-element Boolean vectors as shown in Eq.(3), which we have named BA-2. The use of "or" and "and" between T and F for the states D and X is justified by the descriptions given in Eq.(3). There is no time to give details.

This is not the end of the story, but the beginning of a whole new approach to logic. Perhaps the most significant feature of this approach is the recognition and symbolization of such reverse operators, which lead to what we have called "unary" relations. Thus, the truth tables for OR and AND. can be taken over as 2x2 matrices, as in Eqs.(1a) and (2a) of Fig.25, and the answer to a query such as, "If a AND b is given to be true, and a is given as T (or F), what is b in each case?" can be written in the language of Boolean vector-matrix formalism (BVMF) as in Eqs.(1b,c) and (2b,c). As will be seen from these, all the essential conclusions

Fig. 25: Unary relations in BA-2 algebra and in the general theory of relations.

made in Fig. 24 about the consequences of the reversed operators are reproduced in the examples (1b,c) and (2b,c) by this formalism.

There is no time to talk about various applications of this vector-matrix formalism, using 2-vectors and 3-vectors, for the branches of logic known as propositional calculus and quantifier calculus. I shall, however, indicate the great convenience of the vector-matrix formalism in what is known as predicate calculus, which is most relevant to practical applications of logic. The basic formula that connects two sets of objects can be represented in this formalism by a matrix R_{ij} as shown in Eq.(4). Thus, suppose A_1, A_2 ... A_m are a number of persons and B_1, B_2 ... B_n are the possible cities from which they come. Then, if A_1 belongs to the city B_2, we say that the element R_{12} of the relation "coming from city"is 1, and 0 otherwise. Thus, we obtain an array of R_{ij} ($i = 1$ to m, $j = 1$ to n), each of which is a Boolean number, having only the values 1, 0. Then, if $m = 5$, and we have a subset of A's consisting of A_1, A_3, A_5, then we can represent them by a 5-element Boolean vector $(a_1, a_2, a_3, a_4, a_5) = (1\ 0\ 1\ 0\ 1)$.

BSF Lecture
21.7.87

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}]{\text{Professor of}} \underset{\sim}{p}$$

$$GM1 = \begin{array}{c|cccc} & p_1 & p_2 & p_3 & p_4 \\ \hline s_1 & 1 & 0 & 1 & 0 \\ s_2 & 0 & 1 & 1 & 0 \\ s_3 & 1 & 0 & 0 & 1 \\ s_4 & 0 & 0 & 0 & 1 \\ s_5 & 1 & 1 & 1 & 0 \end{array}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}]{\text{Student of}} \underset{\sim}{s}$$

$$GM3 = \; = GT(GM1) \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$\underset{\sim}{s} \xrightarrow[\underset{\sim}{P}^c]{\text{Not professor of}} \underset{\sim}{p}$$

$$GM2 = \; = GC(GM1) \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\underset{\sim}{p} \xrightarrow[\underset{\sim}{S}^c]{\text{Not student of}} \underset{\sim}{s}$$

$$GM4 = \; = GTC(GM1) \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

1. Who are all the professors who teach the students s_1 and s_2 ?

s_1, s_2 gives GVS = (1 1 0 0 0)	Vector representing given students
GUNPDT(GVS,GM1) = GVP	Relation "student to professor" GM1 applied (1a)
GVP = (1 1 1 0) — Ans: p_1, p_2, p_3	Vector representing required professors. (1b)

2. Which professors among p_2 and p_4 teach at least one of the students s_1, s_3 ?

Input GVP1 = (0 1 0 1), GVS1 = (1 0 1 0 0)

GUNPDT(GVS1,GM1) = GVP, GVP = (1 0 1 1) : p_1, p_3, p_4 teach either s_1 or s_2 (2a)

GVIDYA(GVP,GVP1) = GVP2 , GVP2 = (0 0 0 1) : Of these, only p_4 is contained in the input set p_2, p_4 (2b)

we give some examples in Fig. 27.

The statements there are fairly self-explanatory in view of what has been said earlier. In these problems also, we deal with a set of five students and four professors, but consider two relations that can exist between them, namely $\underset{\sim}{P}$ = "professor of", and $\underset{\sim}{E}$ = "examiner of", whose matrices are given in (1), as GMP and GME. (Note that this GMP is different from that in Fig. 26.) Now we ask two questions as given in (a) and (b) of Fig. 27. It requires some effort to differentiate between the two in common language, but in BVMF, the two are given by quite different formulae, as in (3) and (4) of Fig. 27. Thereafter, we can leave it to the computer, and interestingly, the output vectors GVPA and GVPB, for the questions in (a) and (b), are quite different as given in (5). These mean that the answer to the first problem is that there is only one professor p_4 who satisfies both the conditions of this problem, while the answer to the second problem is that three professors p_1, p_2, p_4 satisfy the given conditions. I am giving this example to indicate how even such tricky problems can be formulated in almost a single line in BVMF, and can be computerized via MATLOG.

Fig. 27: Examples of logical "tongue-twisters" solved using BVMF and MATLOG

As already mentioned in the beginning, this does not at all mean that computers can replace human thought. The computer can only work out the answer to a query regarding the validity of a conclusion in the forms : "yes, "no", "does not follow from the assumption" or "contradictory to known facts"; but it cannot suggest what questions should be asked to clinch the issue, if the deduction is found to be inconclusive. Much can be said on this inter-relationship between semantics and logic, but, in essence, we can say that the latter, which is the grammar of reasoning, can be computerized, while the former continues to be still in the realm of human endeavour.

we give some examples in Fig. 27.

The statements there are fairly self-explanatory in view of what has been said earlier. In these problems also, we deal with a set of five students and four professors, but consider two relations that can exist between them, namely $\underset{\sim}{P}$ = "professor of", and $\underset{\sim}{E}$ = "examiner of", whose matrices are given in (1), as GMP and GME. (Note that this GMP is different from that in Fig. 26.) Now we ask two questions as given in (a) and (b) of Fig. 27. It requires some effort to differentiate between the two in common language, but in BVMF, the two are given by quite different formulae, as in (3) and (4) of Fig. 27. Thereafter, we can leave it to the computer, and interestingly, the output vectors GVPA and GVPB, for the questions in (a) and (b), are quite different as given in (5). These mean that the answer to the first problem is that there is only one professor p_4 who satisfies both the conditions of this problem, while the answer to the second problem is that three professors p_1, p_2, p_4 satisfy the given conditions. I am giving this example to indicate how even such tricky problems can be formulated in almost a single line in BVMF, and can be computerized <u>via</u> MATLOG.

Fig. 27: Examples of logical "tongue-twisters" solved using BVMF and MATLOG

As already mentioned in the beginning, this does not at all mean that computers can replace human thought. The computer can only work out the answer to a query regarding the validity of a conclusion in the forms : "yes, "no", "does not follow from the assumption" or "contradictory to known facts"; but it cannot suggest what questions should be asked to clinch the issue, if the deduction is found to be inconclusive. Much can be said on this inter-relationship between semantics and logic, but, in essence, we can say that the latter, which is the grammar of reasoning, can be computerized, while the former continues to be still in the realm of human endeavour.